

带你了解2D动画世界



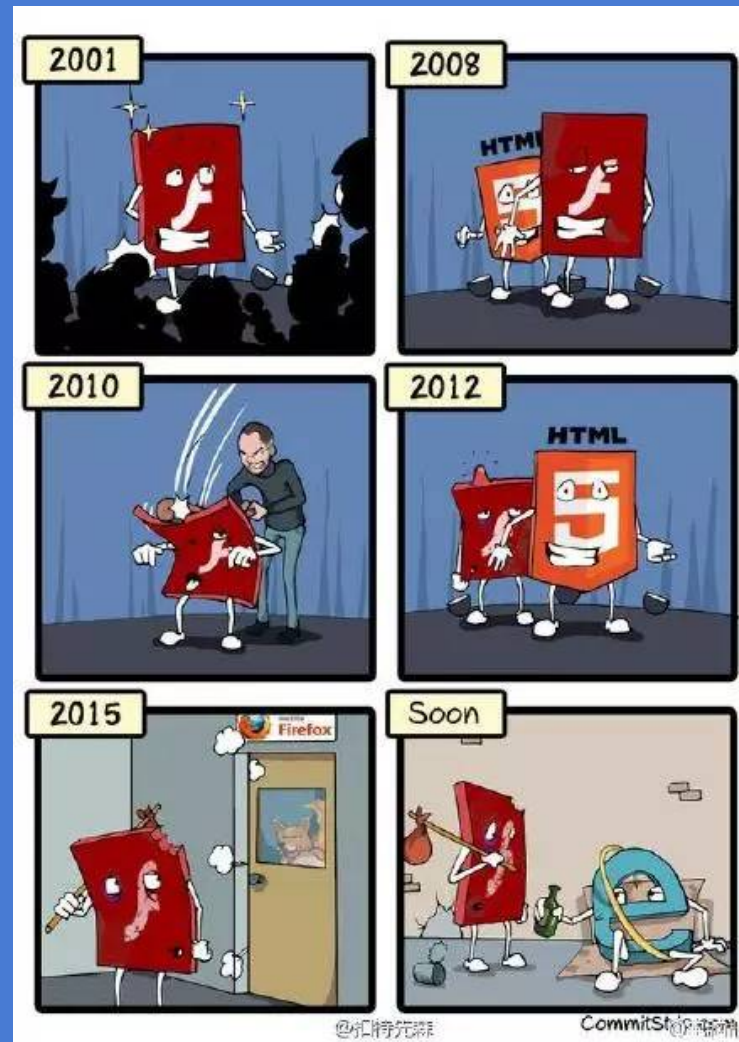
张百鸽

美小技术部 - 前端组

刀耕火种的前端史

年份	事件
1994	Tim Berners – Lee(万维网之父)创建W3C，Tim基友发布CSS 网景推出了第一版Navigator 前端起点(Web 1.0时代)
2003	SVG于2003年1月14日成为W3C推荐标准,SVG 的主要竞争者是 Flash,SVG可缩放矢量图形
2005	谷歌利用Ajax技术打造Gmail和谷歌地图之后 – Ajax大受关注 前端的第一次飞跃（进入Web2.0时代）
2010	推出html5的画布功能-Canvas,在网页上绘制2D图像 canvas 元素绘制路径、矩形、圆形、字符以及添加图像
2014	HTML5标准发布，支持网页端的Audio、Video等多媒体功能，基于SVG、Canvas、WebGL及CSS3的3D功能，Html5取代Flash在移动设备的地位
2015	Flash正式走向消亡，Adobe公司推出Animate CC取代之之前Flash Professional CC；支持Flash SWF、AIR格式的同时，还会支持HTML5 Canvas、WebG，并能通过可扩展架构去支持包括SVG在内的几乎任何动画格式。

Flash的没落



2D的世界

2D动画	是指二维交互式动画， 也就是我们通常所说的2d游戏种的美术资源（人物行走、人物状态、地图等等）都是以png或jpg的图形文件渲染而成，2d是没办法完成视角转换（现在比较流行的2D设计形式：扁平化、MEB等）
间动画和帧动画	区间动画：从A到B执行的过度动画； 逐帧动画：从A到B，从B到C的连续的动画播放。 其中动画标准是一秒24帧，也就是24张画面组成一秒。电视播放标准是25帧每秒（但是12） 关键帧：能够叙述故事发展的重要帧 中间帧：过度帧，补充重要帧之间，使动画连续 极端帧:极端帧不一定是关键帧，是两个关键帧之间的次重要
2D动画技术	SVG、Canvas、CSS3

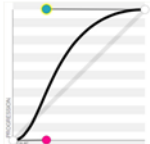
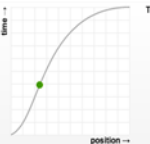

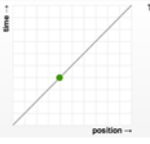
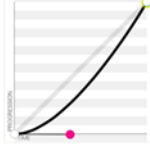

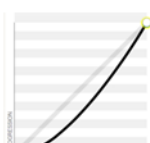
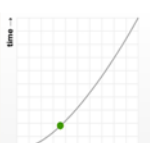
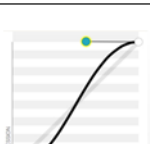

-	-
2D动画js框	jQuery Transit、Transform.js、Move.js、Velocity.js、vivus.js、snabbt.js

带你了解CSS3的世界

(transform)	2D旋转: rotate旋转 \ 2D旋转: scale缩放 \ skew扭曲 \ translate移动 \ matrix矩阵变形
过度 (transition)	property duration timing-function delay(all 0 ease 0)
动画 (animation)	animation-name \ animation-duration \ animation-delay \ animation-iteration-count(infinite) \ animation-direction \ animation-play-state \ animation-fill-mode \ animation-timing-function(控制时间函数-速度曲线函数-缓动函数)
animation-timing-function	linear、ease-in、ease-out、ease-in-out、cubic-bezier(贝塞尔曲线)
Keyframes	固定的语法规则@keyframes *name { 0% {} 100% {} }

-	-
Demo	http://www.html5tricks.com/demo/html5-boy-animation/index.html http://www.html5tricks.com/demo/css3-shake-head-animation/index.html

timing-function

函数	功能描述	图例																								
ease	默认值, 元素样式从初始状态过渡到终止状态时速度由快到慢, 逐渐变慢	  <table><thead><tr><th>Time (s)</th><th>Position</th></tr></thead><tbody><tr><td>0.000</td><td>0%</td></tr><tr><td>0.500</td><td>9%</td></tr><tr><td>1.000</td><td>29%</td></tr><tr><td>1.500</td><td>51%</td></tr><tr><td>2.000</td><td>69%</td></tr><tr><td>2.500</td><td>81%</td></tr><tr><td>3.000</td><td>89%</td></tr><tr><td>3.500</td><td>94%</td></tr><tr><td>4.000</td><td>98%</td></tr><tr><td>4.500</td><td>99%</td></tr><tr><td>5.000</td><td>100%</td></tr></tbody></table>	Time (s)	Position	0.000	0%	0.500	9%	1.000	29%	1.500	51%	2.000	69%	2.500	81%	3.000	89%	3.500	94%	4.000	98%	4.500	99%	5.000	100%
Time (s)	Position																									
0.000	0%																									
0.500	9%																									
1.000	29%																									
1.500	51%																									
2.000	69%																									
2.500	81%																									
3.000	89%																									
3.500	94%																									
4.000	98%																									
4.500	99%																									
5.000	100%																									
linear	元素样式从初始状态过渡到终止状态速度是恒速	  <table><thead><tr><th>Time (s)</th><th>Position</th></tr></thead><tbody><tr><td>0.000</td><td>0%</td></tr><tr><td>0.500</td><td>10%</td></tr><tr><td>1.000</td><td>20%</td></tr><tr><td>1.500</td><td>30%</td></tr><tr><td>2.000</td><td>40%</td></tr><tr><td>2.500</td><td>50%</td></tr><tr><td>3.000</td><td>60%</td></tr><tr><td>3.500</td><td>70%</td></tr><tr><td>4.000</td><td>80%</td></tr><tr><td>4.500</td><td>90%</td></tr><tr><td>5.000</td><td>100%</td></tr></tbody></table>	Time (s)	Position	0.000	0%	0.500	10%	1.000	20%	1.500	30%	2.000	40%	2.500	50%	3.000	60%	3.500	70%	4.000	80%	4.500	90%	5.000	100%
Time (s)	Position																									
0.000	0%																									
0.500	10%																									
1.000	20%																									
1.500	30%																									
2.000	40%																									
2.500	50%																									
3.000	60%																									
3.500	70%																									
4.000	80%																									
4.500	90%																									
5.000	100%																									
ease-in	元素样式从初始状态过渡到终止状态时, 速度越来越快, 呈一种加速状态。常称这种效果为渐显效果	  <table><thead><tr><th>Time (s)</th><th>Position</th></tr></thead><tbody><tr><td>0.000</td><td>0%</td></tr><tr><td>0.500</td><td>2%</td></tr><tr><td>1.000</td><td>6%</td></tr><tr><td>1.500</td><td>13%</td></tr><tr><td>2.000</td><td>21%</td></tr><tr><td>2.500</td><td>32%</td></tr><tr><td>3.000</td><td>43%</td></tr><tr><td>3.500</td><td>55%</td></tr><tr><td>4.000</td><td>69%</td></tr><tr><td>4.500</td><td>84%</td></tr><tr><td>5.000</td><td>100%</td></tr></tbody></table>	Time (s)	Position	0.000	0%	0.500	2%	1.000	6%	1.500	13%	2.000	21%	2.500	32%	3.000	43%	3.500	55%	4.000	69%	4.500	84%	5.000	100%
Time (s)	Position																									
0.000	0%																									
0.500	2%																									
1.000	6%																									
1.500	13%																									
2.000	21%																									
2.500	32%																									
3.000	43%																									
3.500	55%																									
4.000	69%																									
4.500	84%																									
5.000	100%																									
ease-out	元素样式从初始状态过渡到终止状态时, 速度越来越慢, 呈一种减速状态。常称这种效果为渐隐效果	  <table><thead><tr><th>Time (s)</th><th>Position</th></tr></thead><tbody><tr><td>0.000</td><td>0%</td></tr><tr><td>0.500</td><td>2%</td></tr><tr><td>1.000</td><td>6%</td></tr><tr><td>1.500</td><td>13%</td></tr><tr><td>2.000</td><td>21%</td></tr><tr><td>2.500</td><td>32%</td></tr><tr><td>3.000</td><td>43%</td></tr><tr><td>3.500</td><td>55%</td></tr><tr><td>4.000</td><td>69%</td></tr><tr><td>4.500</td><td>84%</td></tr><tr><td>5.000</td><td>100%</td></tr></tbody></table>	Time (s)	Position	0.000	0%	0.500	2%	1.000	6%	1.500	13%	2.000	21%	2.500	32%	3.000	43%	3.500	55%	4.000	69%	4.500	84%	5.000	100%
Time (s)	Position																									
0.000	0%																									
0.500	2%																									
1.000	6%																									
1.500	13%																									
2.000	21%																									
2.500	32%																									
3.000	43%																									
3.500	55%																									
4.000	69%																									
4.500	84%																									
5.000	100%																									
ease-in-out	元素样式从初始状态到终止状态时, 先加速再减速。常称这种效果为渐显渐隐效果	  <table><thead><tr><th>Time (s)</th><th>Position</th></tr></thead><tbody><tr><td>0.000</td><td>0%</td></tr><tr><td>0.500</td><td>2%</td></tr><tr><td>1.000</td><td>6%</td></tr><tr><td>1.500</td><td>19%</td></tr><tr><td>2.000</td><td>33%</td></tr><tr><td>2.500</td><td>50%</td></tr><tr><td>3.000</td><td>67%</td></tr><tr><td>3.500</td><td>81%</td></tr><tr><td>4.000</td><td>92%</td></tr><tr><td>4.500</td><td>98%</td></tr><tr><td>5.000</td><td>100%</td></tr></tbody></table>	Time (s)	Position	0.000	0%	0.500	2%	1.000	6%	1.500	19%	2.000	33%	2.500	50%	3.000	67%	3.500	81%	4.000	92%	4.500	98%	5.000	100%
Time (s)	Position																									
0.000	0%																									
0.500	2%																									
1.000	6%																									
1.500	19%																									
2.000	33%																									
2.500	50%																									
3.000	67%																									
3.500	81%																									
4.000	92%																									
4.500	98%																									
5.000	100%																									

带你了解Velocity.js

Velocity 是一个简单易用、高性能、功能丰富的轻量级JS动画库。它能和 jQuery 完美协作，并和\$.animate()有相同的 API，但它不依赖 jQuery，可单独使用。Velocity 不仅包含了 \$.animate() 的全部功能，还拥有：颜色动画、转换动画(transforms)、循环、缓动、SVG 动画、和 滚动动画 等特色功能。

Velocity.js用法

	-	-	
	依赖	JQuery、Velocity、Velocity.UI	
	用法	<code>\$ele.velocity({ width: 100 }, { duration: 300 })</code>	

```
$element.velocity({
  width: "100px",           // 动画属性 宽度到 "500px" 的动画
  * property2: value2       // 属性示例
}, {
  /* Velocity 动画配置项的默认值 */
  * duration: 300,          // 动画执行时间
  * easing: "swing",        // 缓动效果
  queue: "",                // 队列
  begin: undefined,         // 动画开始时的回调函数
  * progress: undefined,    // 动画执行中的回调函数 (该函数会随着动画执行被不断触发)
  * complete: undefined,   // 动画结束时的回调函数
  display: undefined,       // 动画结束时设置元素的 css display 属性
  visibility: undefined,   // 动画结束时设置元素的 css visibility 属性
  loop: false,              // 循环
  * delay: false,           // 延迟
  mobileHA: true            // 移动端硬件加速 (默认开启)
});
```


Velocity的链式动画

```
$element
```

```
/* 先执行宽度变为75px的动画 */
```

```
.velocity({ width: 75 })
```

```
/* 等前面的宽度动画结束后, 再执行高度变为0的动画 */
```

```
.velocity({ height: 0 });
```

Velocity的时间轴动画

```
const enterAn = [{
  elements: cloud1,
  properties: 'transition.slideUpBigIn',
  options: { delay: 300, complete: function() { cloud1.addClass('shake-cloud1'); } },
}, {
  elements: cloud2,
  properties: 'transition.slideUpBigIn',
  options: { sequenceQueue: false, complete: function() { cloud2.addClass('shake-cl
}, {
  elements: cloud3,
  properties: 'transition.slideDownBigIn',
  options: { sequenceQueue: false, complete: function() { cloud3.addClass('shake-cl
}, {
  elements: sun,
  properties: 'transition.slideRightBigIn',
  options: { sequenceQueue: false, complete: function() { sun.addClass('uniformlyRo
}];
$.Velocity.RunSequence(enterAn);
```

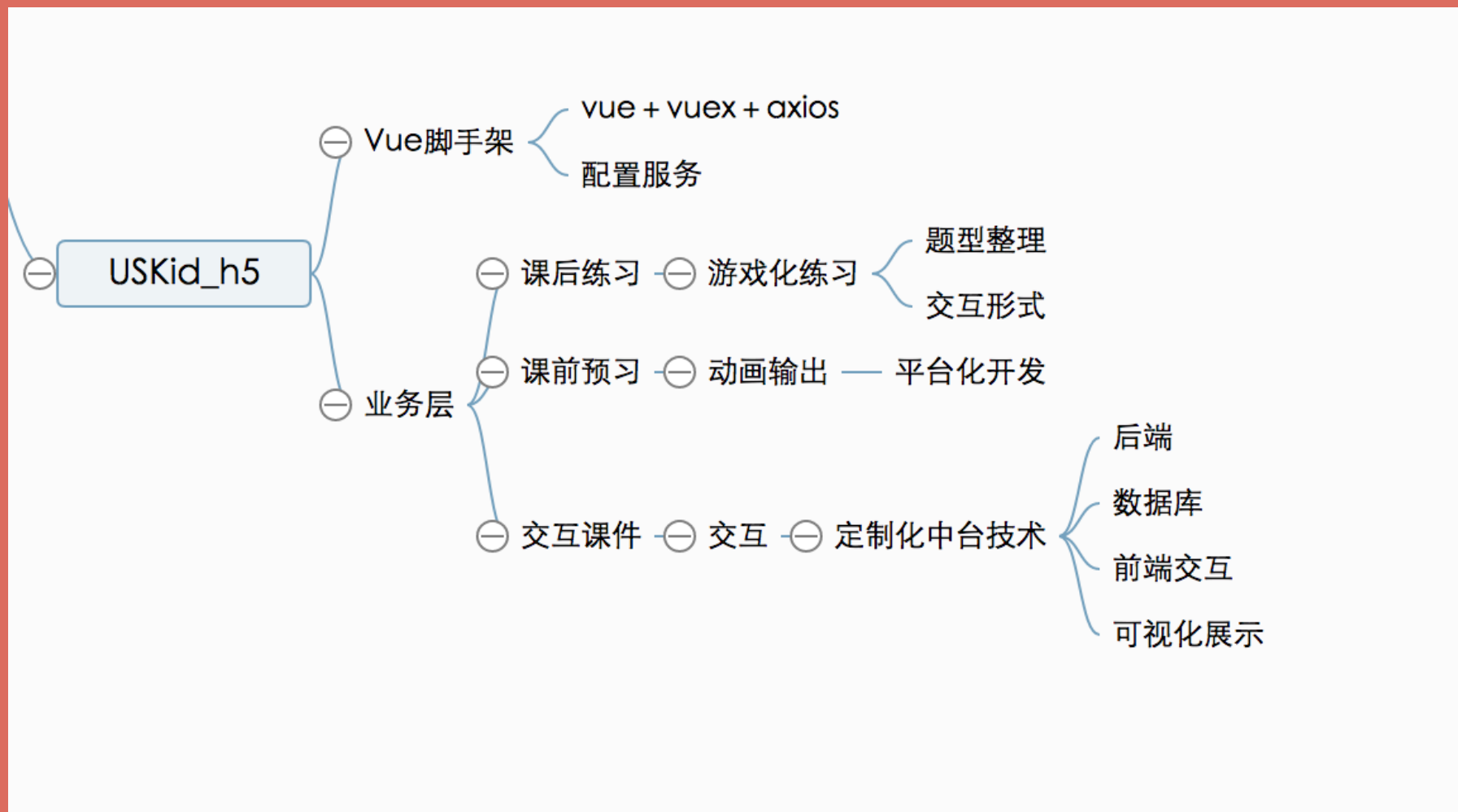
动画开发流程

-	-
需求分析	需要的动画场景和素材整理
素材整理	涉及到帧动画的图片素材，需要对其进行定位
代码编写	定位动画静态元素，添加不需要事件驱动的CSS3动画；添加需要事件驱动的JS动画（连续动画，帧动画）
Demo演示	http://localhost:8080/#/hello

额外的内容

- USKid-h5项目
- C线的述职

USKid-h5



C线述职

