JavaScript引擎的性能优化

北京奇虎科技有限公司
任寰

提纲

- **JavaScript引擎优化的主要方法**
  - 导出类（hidden class）
  - 内联（Inline cache）
  - JIT
  - 垃圾回收
  - 堆快照
  - 按需优化编译
- **最新进展**
  - 世界之窗
  - Dart

优化JavaScript的难点

- **动态类型语言**
  - 对象具备属性
  - 可以随时增减对象的属性
  - 不同属性的对象可以出现在同样的调用中
  - 原型（prototype）链
  - 函数可以从一个对象移到另一个对象

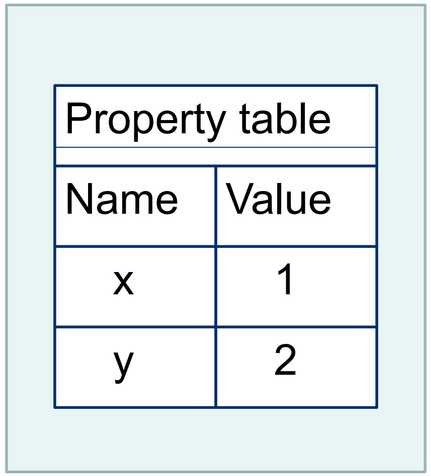
动态类型语言

```
var p = { };
p.x = 1;
p.y = 2;

p
Object {x: 1, y: 2}
```

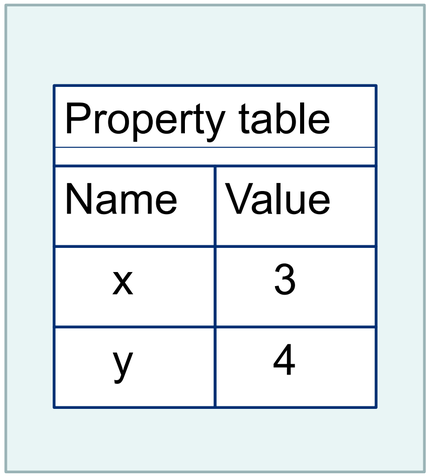
```
function Point(x, y) {
    this.x = x;
    this.y = y;
}

var p = new Point(1,2);
var q = new Point(3,4);
```

动态语言中类的实现

Object p

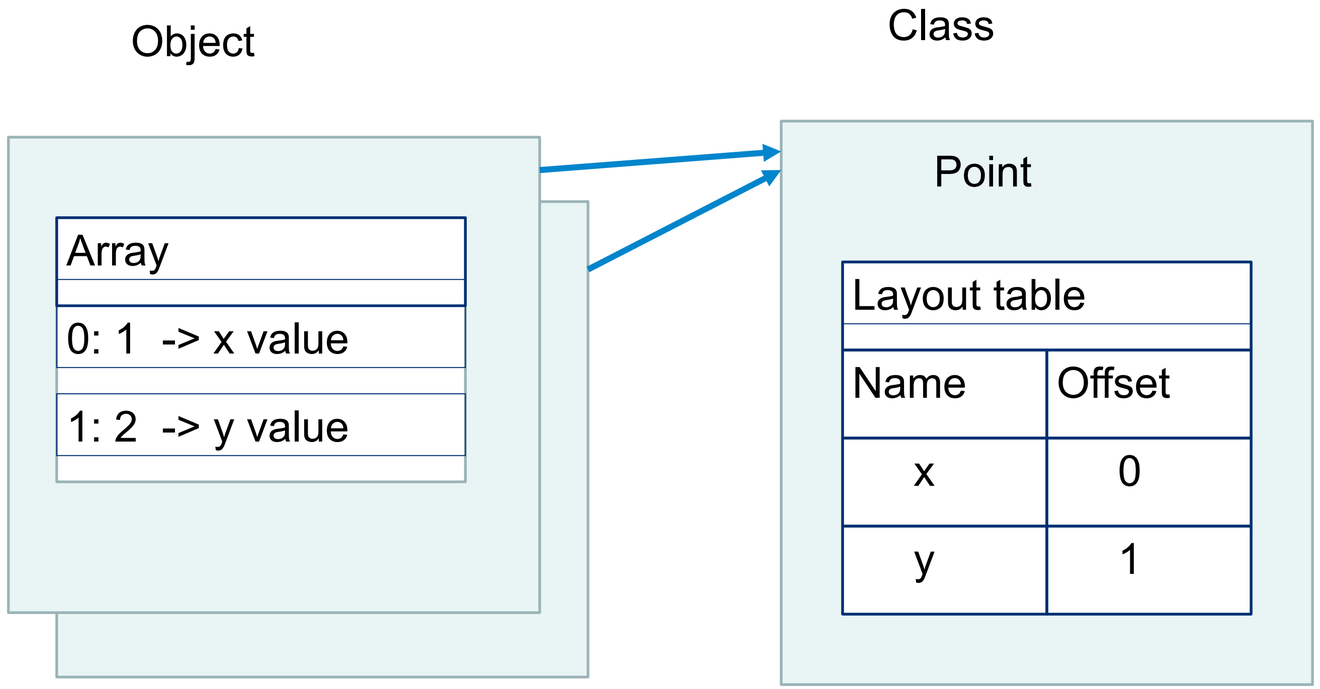


Object q



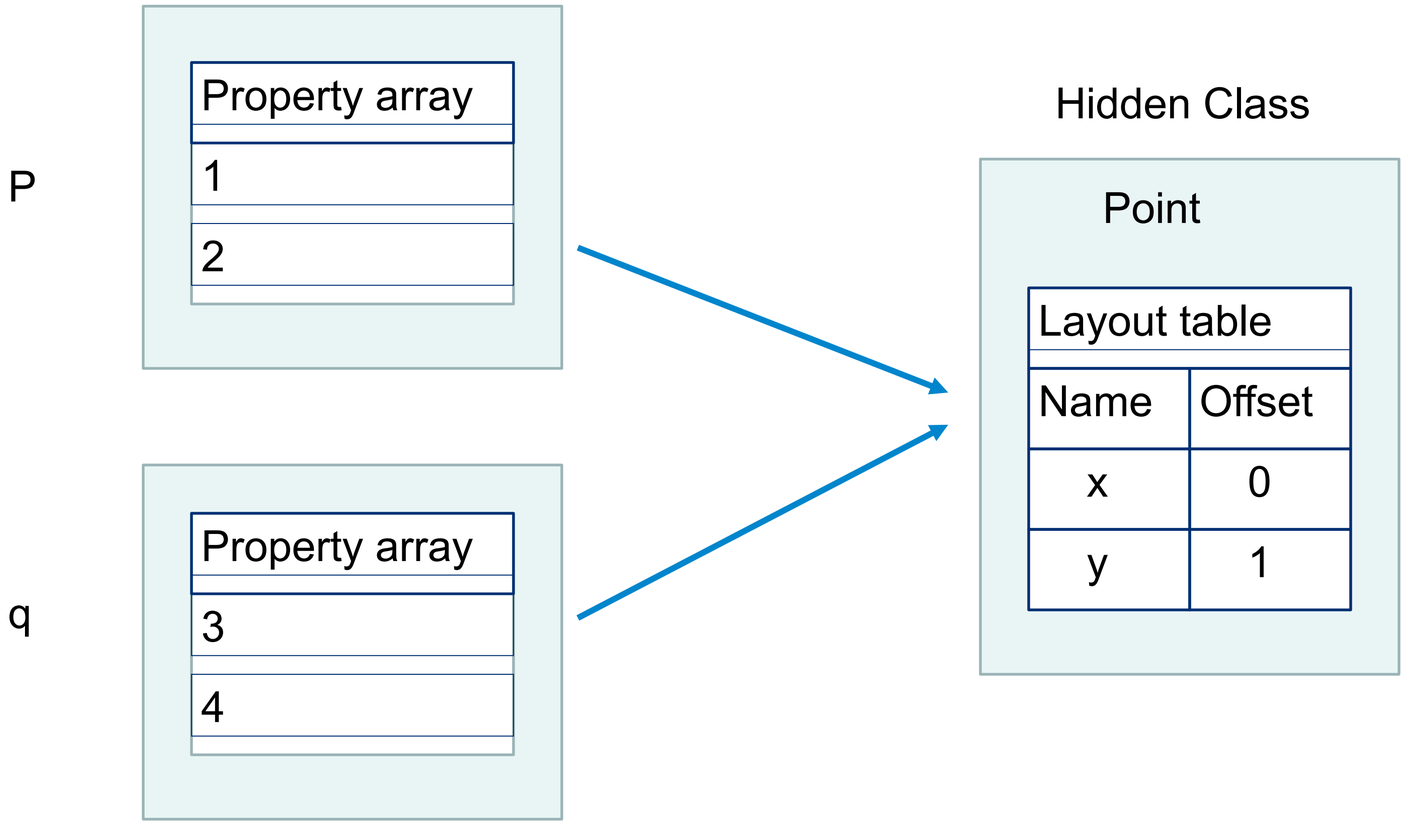
存取对象属性：字典查找（Hash table）

强类型



存取对象的成员：数组 + 位移

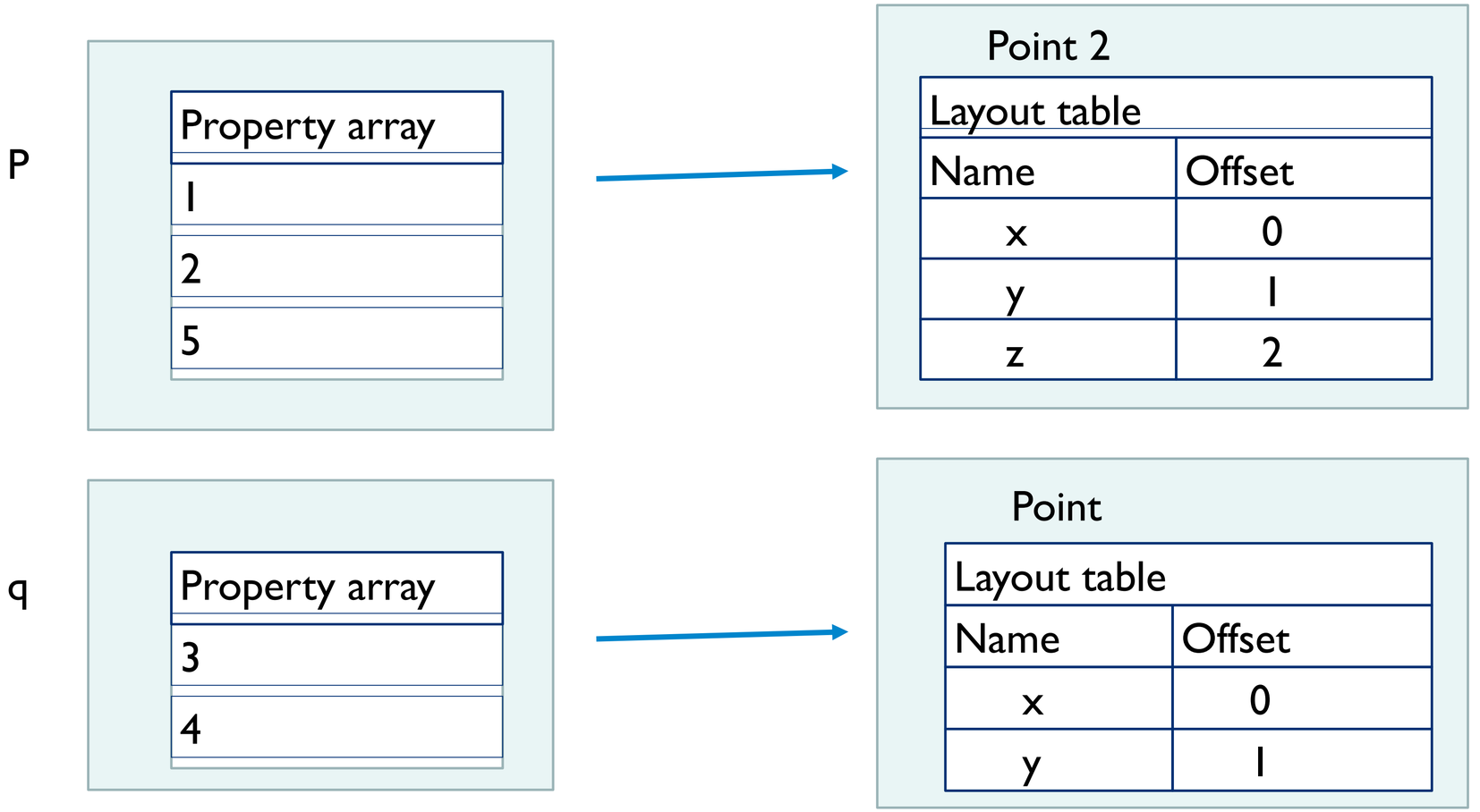
隐藏类



类型转换

```
function Point(x, y) {
    this.x = x;
    this.y = y;
}

var p = new Point(1,2);
var q = new Point(3,4);

p.z = 5;
```

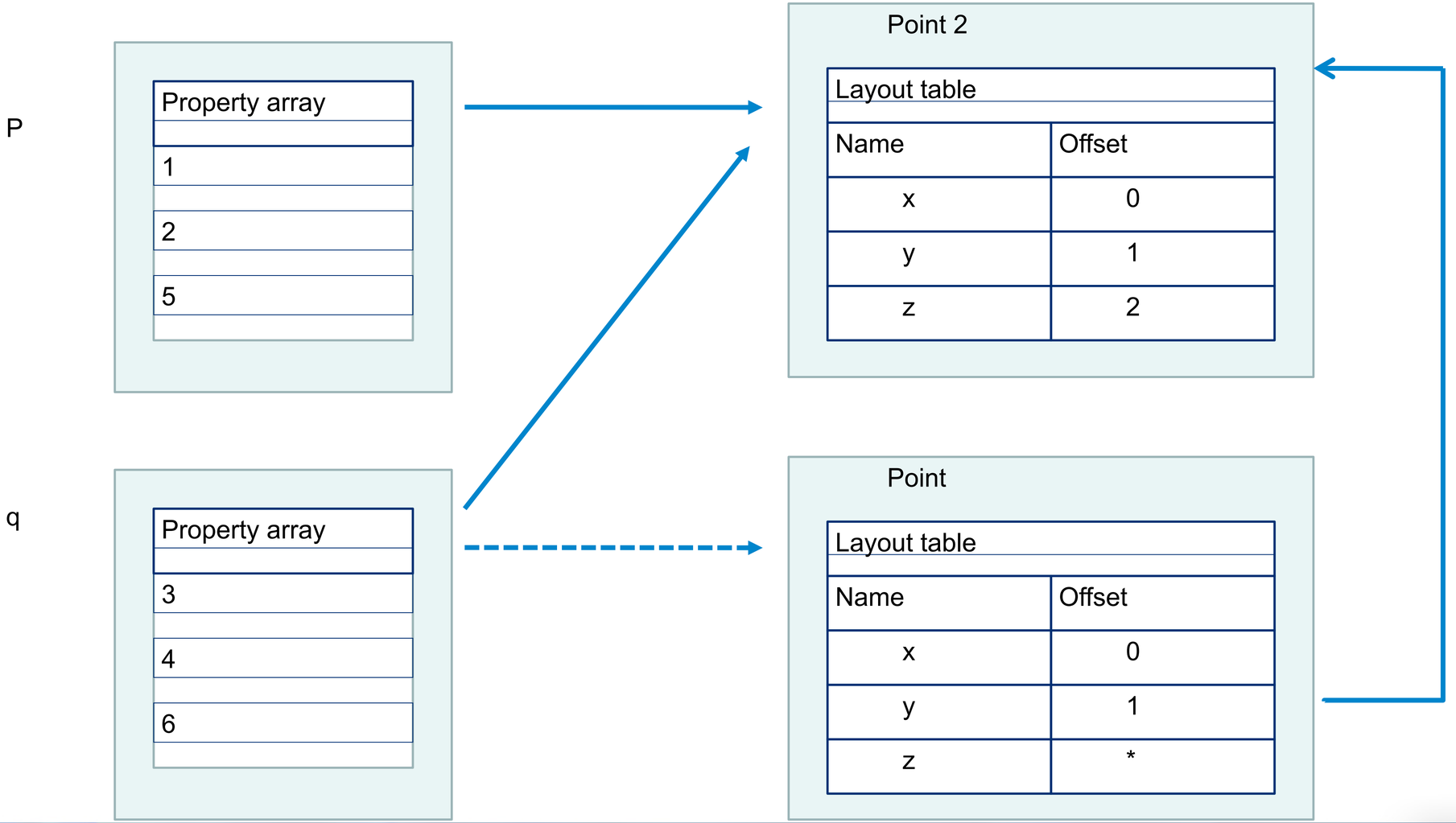
类型转换



类型转换

```
function Point(x, y) {
    this.x = x;
    this.y = y;
}

var p = new Point(1,2);
var q = new Point(3,4);

p.z = 5;
q.z = 6;
```

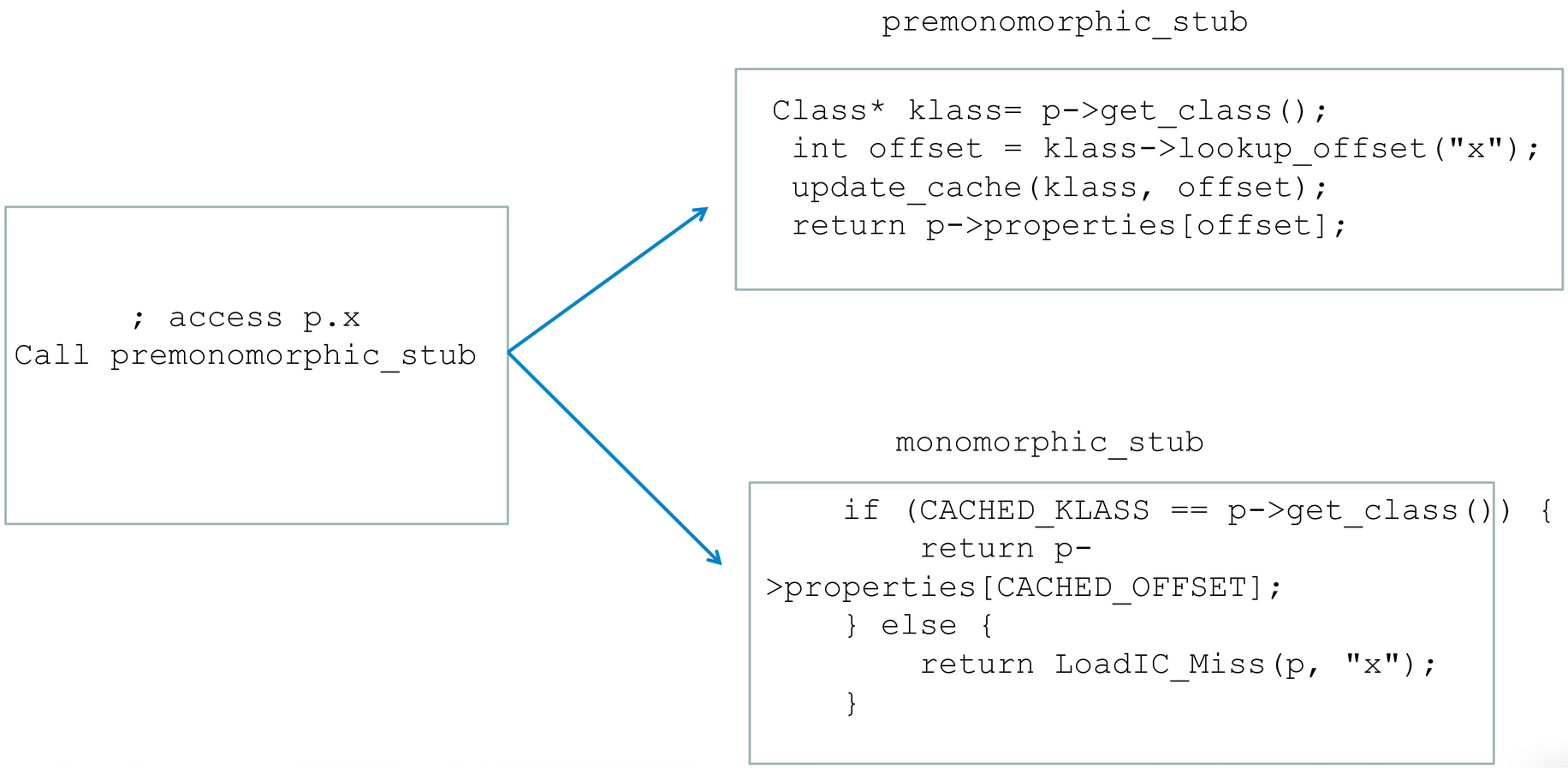
类型转换



优化效果

- **90%的对象使用同一个隐藏类。**
- 影响因素
  - 创建隐藏类的开销
  - 更新隐藏类的开销
- 建议
  - 在构造函数中初始化对象
  - 按同样的顺序初始化

内联（Inline Cache）



Monomorphic Stub

```
0xf7c0d32d: [Code]
Instructions (size = 37)
0xf7c0d344    0  8b442404        mov eax,[esp+0x4]
0xf7c0d348    4  a801            test al,0x1
0xf7c0d34a    6  0f8414000000    jz 32
0xf7c0d350   12  8178ff81ab8ff7 cmp [eax+0xff],0xf78fab81
0xf7c0d357   19  0f8507000000    jnz 32
0xf7c0d35d   25  8b5803          mov ebx,[eax+0x3]
0xf7c0d360   28  8b4307          mov eax,[ebx+0x7]
0xf7c0d363   31  c3              ret
0xf7c0d364   32  e993daffff      jmp LoadIC_Miss
```

存取对象属性的速度提高～11倍。

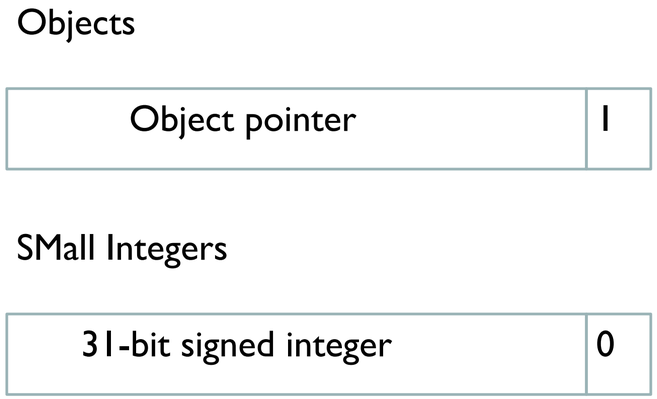
JIT编译

- **JSC（JavaScriptCore）的演进**
  - 源码 → 语法树 → 解释器
  - 源码 → byte code → 解释器 （SquirrelFish）
  - 源码 → byte code → JIT → 执行（SF Extreme）
- **V8**
  - 源码 → JIT → 执行

垃圾回收

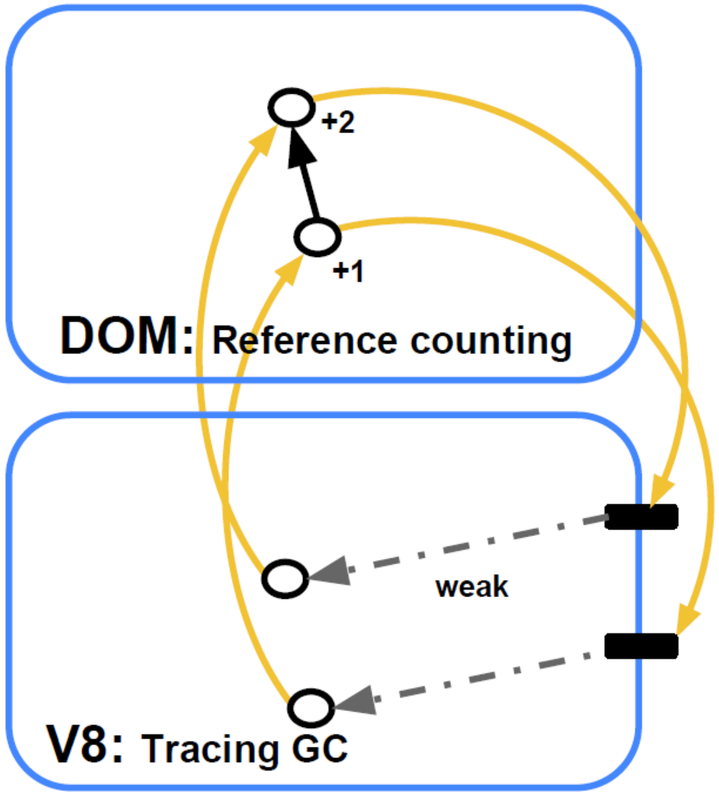
- 精准回收（**precise garbage collection**）
  - 分代式回收（generational）

- 标记（**Tagged Values**）



- 建议：尽量进行2^30内的数值运算

垃圾回收和DOM



DOM对象不能有循环依赖。

图片来源：Google

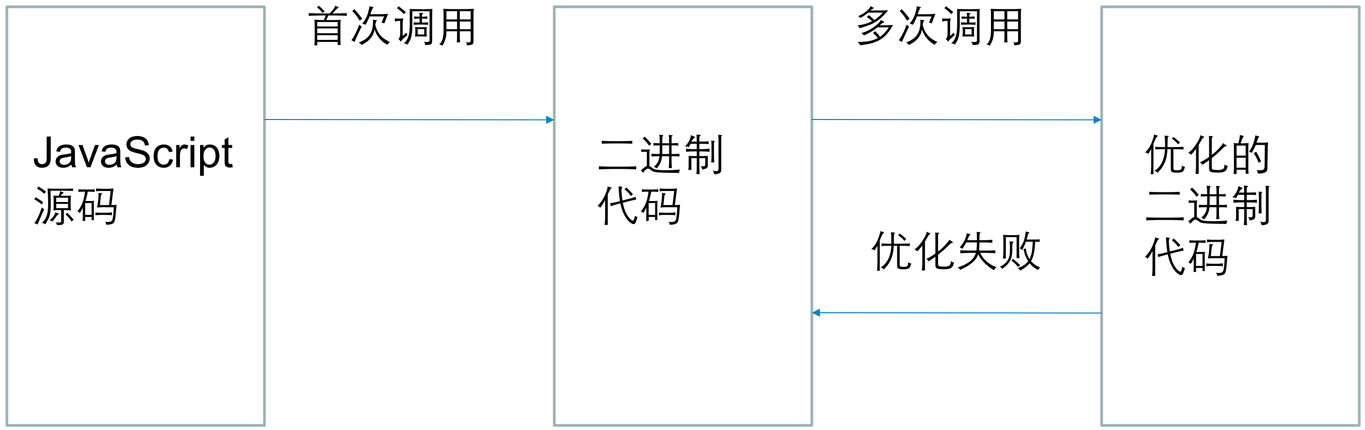
JavaScript库

- **JavaScript的标准库（ECMA 262）用JavaScript实现**
- 优点
  - 开发速度，维护成本
  - 发掘V8引擎的优势
- 缺点
  - 初始化：30ms

堆快照

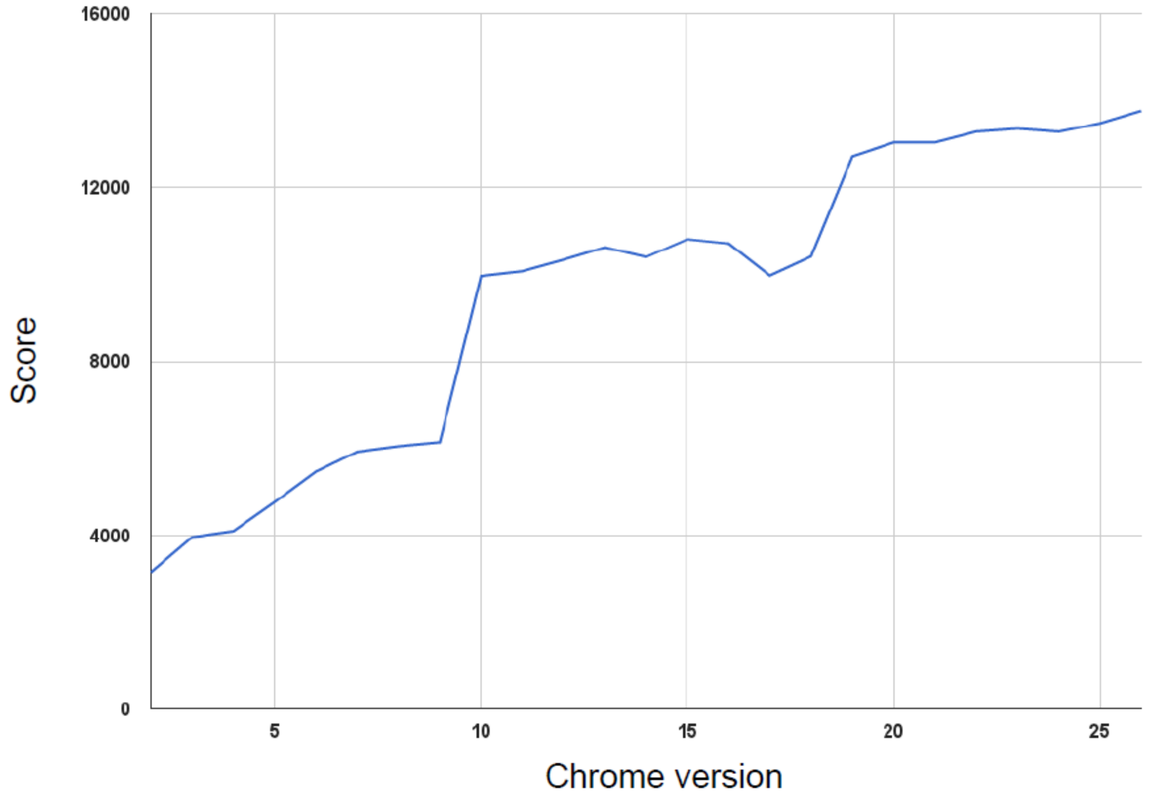
- 用快照（**snapshot**）直接初始化堆（**object heap**）
  - 快速序列化
  - 内置编译好的JavaScript库
  - 编译时生成快照。
- 启动时间：**4-8ms**

自适应编译



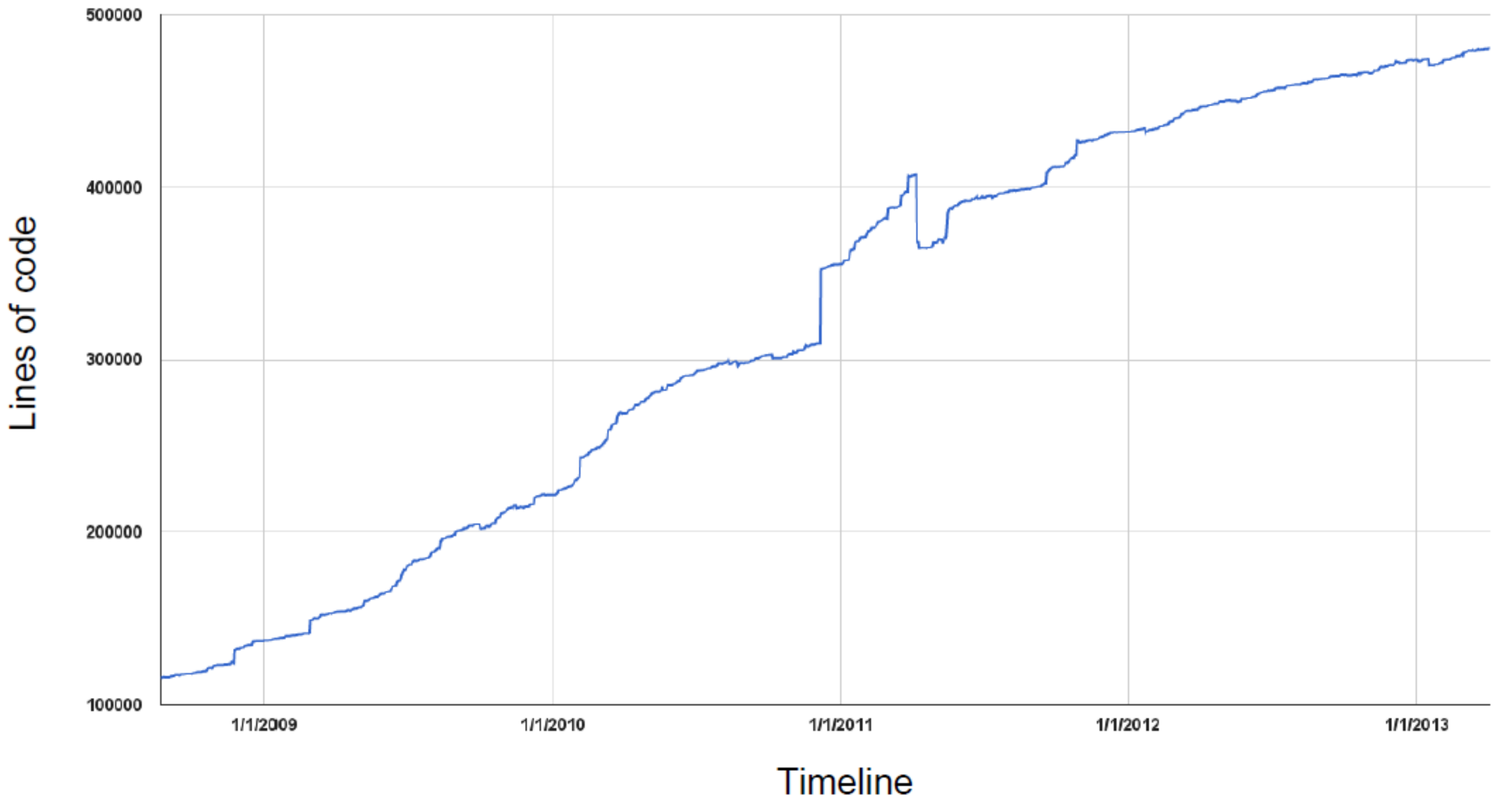
Inline Caches

性能提升历程



数据来源：Chrome V8 团队

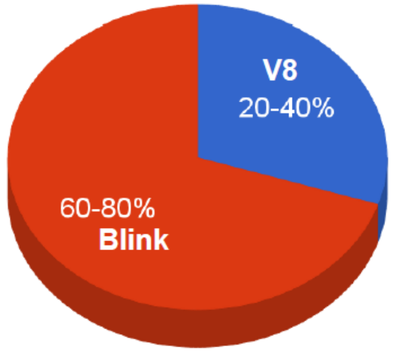
代码复杂度

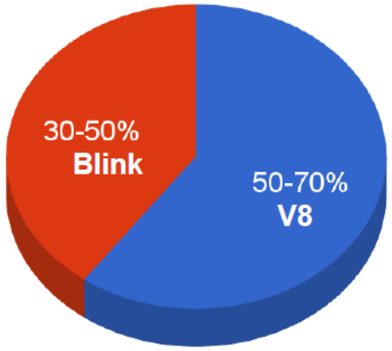


数据来源：Chrome V8 团队

还需要再快吗?

Twitter
Facebook
YouTube
Wikipedia





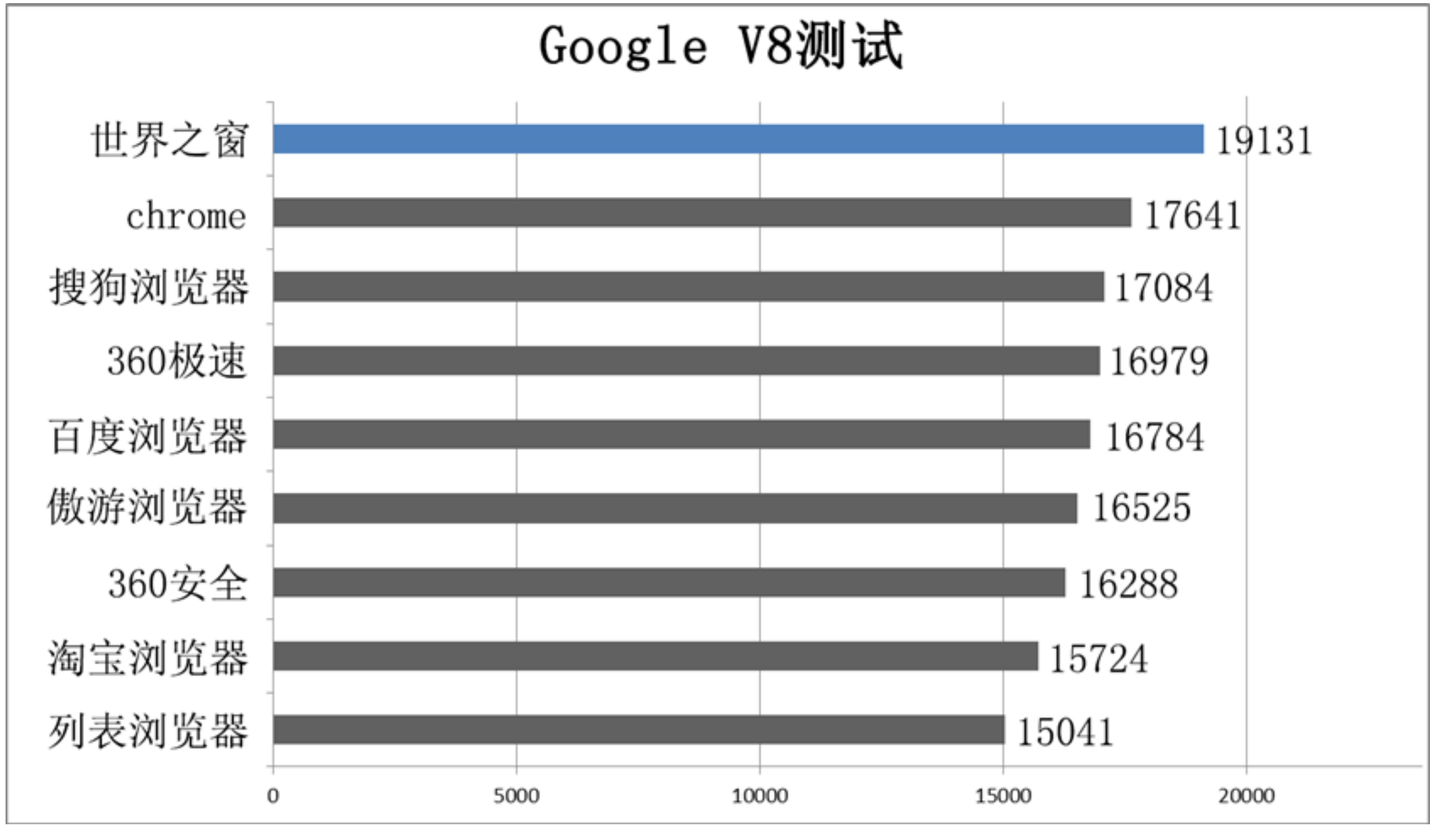
Gmail
Google Docs
Google Search
Google Calendar

数据来源：Chrome V8 团队

还能再快吗?

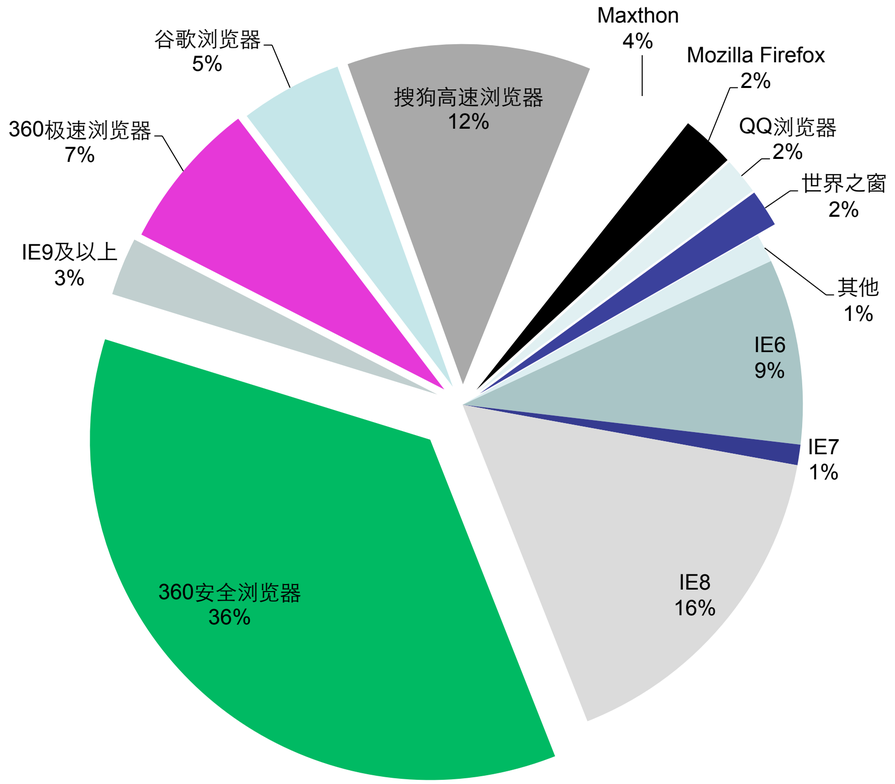
- **Blink/V8**
  - DOM性能与内存管理
- **世界之窗6.0**
  - JavaScript库
  - VM自身的代码优化
- **突破JavaScript语言的局限**
  - Dart

世界之窗6.0


Google V8测试
世界之窗
chrome
搜狗浏览器
360极速
百度浏览器
傲游浏览器
360安全
淘宝浏览器
列表浏览器
19131
17641
17084
16979
16784
16525
16288
15724
15041
0
5000
10000
15000
20000

DART

- **Dart语言**
  - JavaScript语法
  - 更简单的对象模型
- **Dart2js**
  - 生成更利于优化的JavaScript代码



2009 2012 2013

为明天而优化

Chrome 双核浏览器 360SE 6.0 世界之窗 6.0

谢谢！