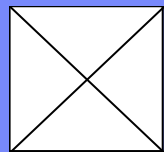
Advanced web technology

Web高级开发与应用技术

web2.0与相关技术



Web2.0

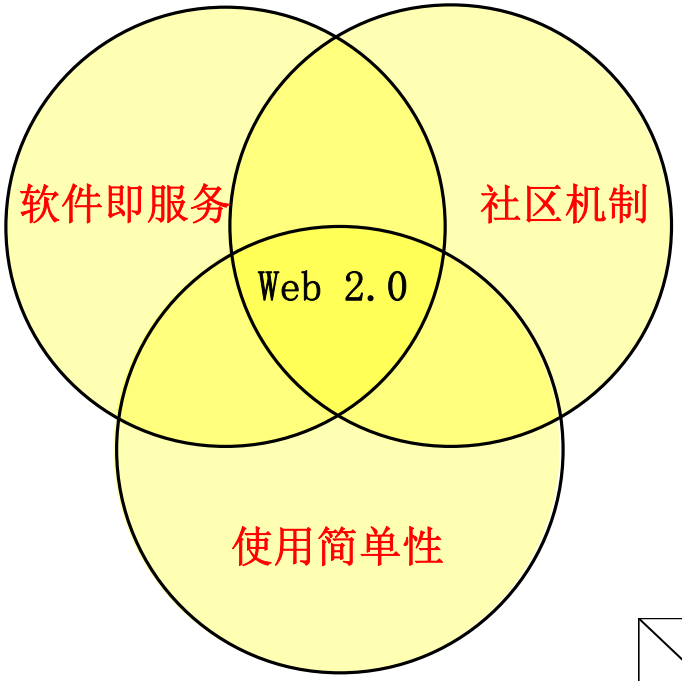
- 最初是由O
- 概念

We
为
实

- 特征
  - Web2.0,
    联网应用
    纯通过网
    具性更强

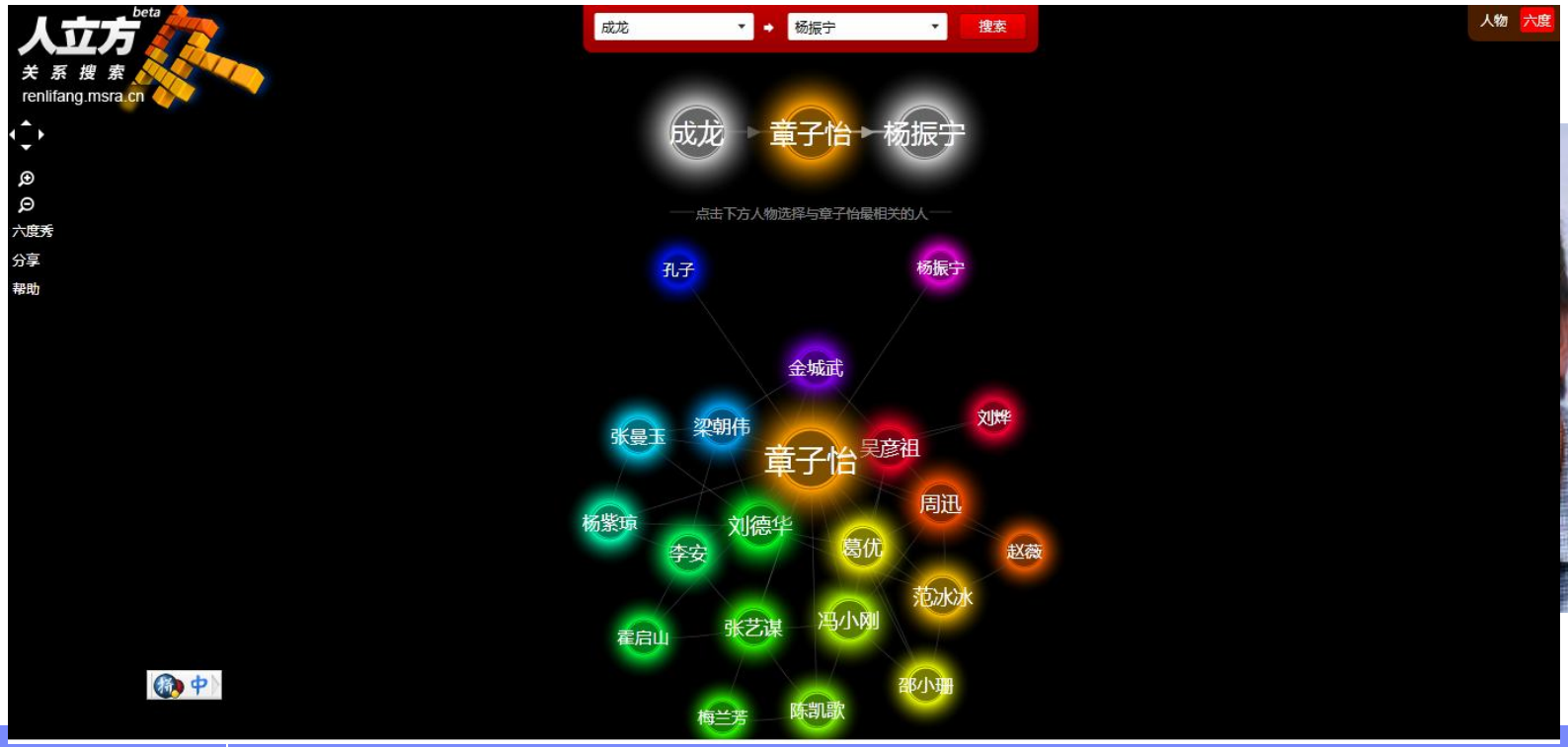
Web2.0

- **Web 2.0的特质**
  - 大规模互连： 网络效应使得边际同核心一样重要
  - 去中心化
  - 以用户为中心
  - 开放
  - 轻量级
  - 自然浮现： 成功来自合作，而不是控制
  - 对软件的影响



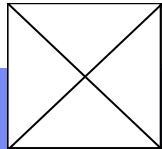
Web2.0

- 六度关系理论
  - 20世纪60年代由美国的心理学家米格兰姆（Stanley Milgram）提出
  - 最多通过六个人你就能够认识任何一个陌生人
  - What Jure Leskovec found:



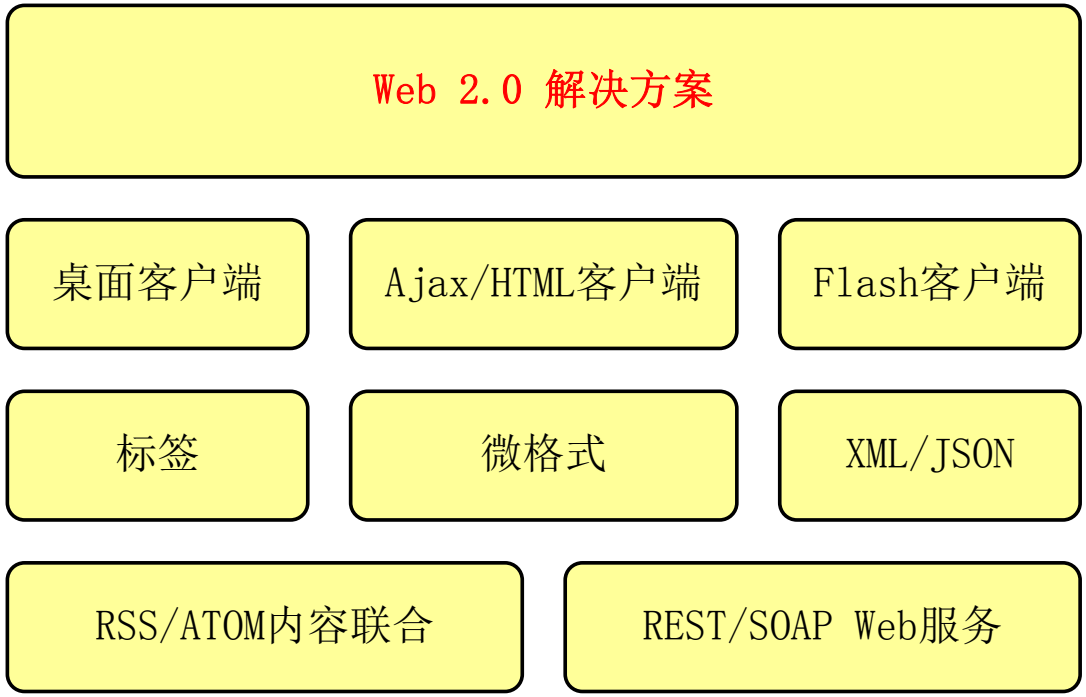
Web2.0

- 微内容
  - 英文是microcontent。用户所生产的任何数据都算是微内容
  - Web 2.0的产品/服务：服务于用户个体的微内容的收集、创建、发布、管理、分享、合作、维护等的平台

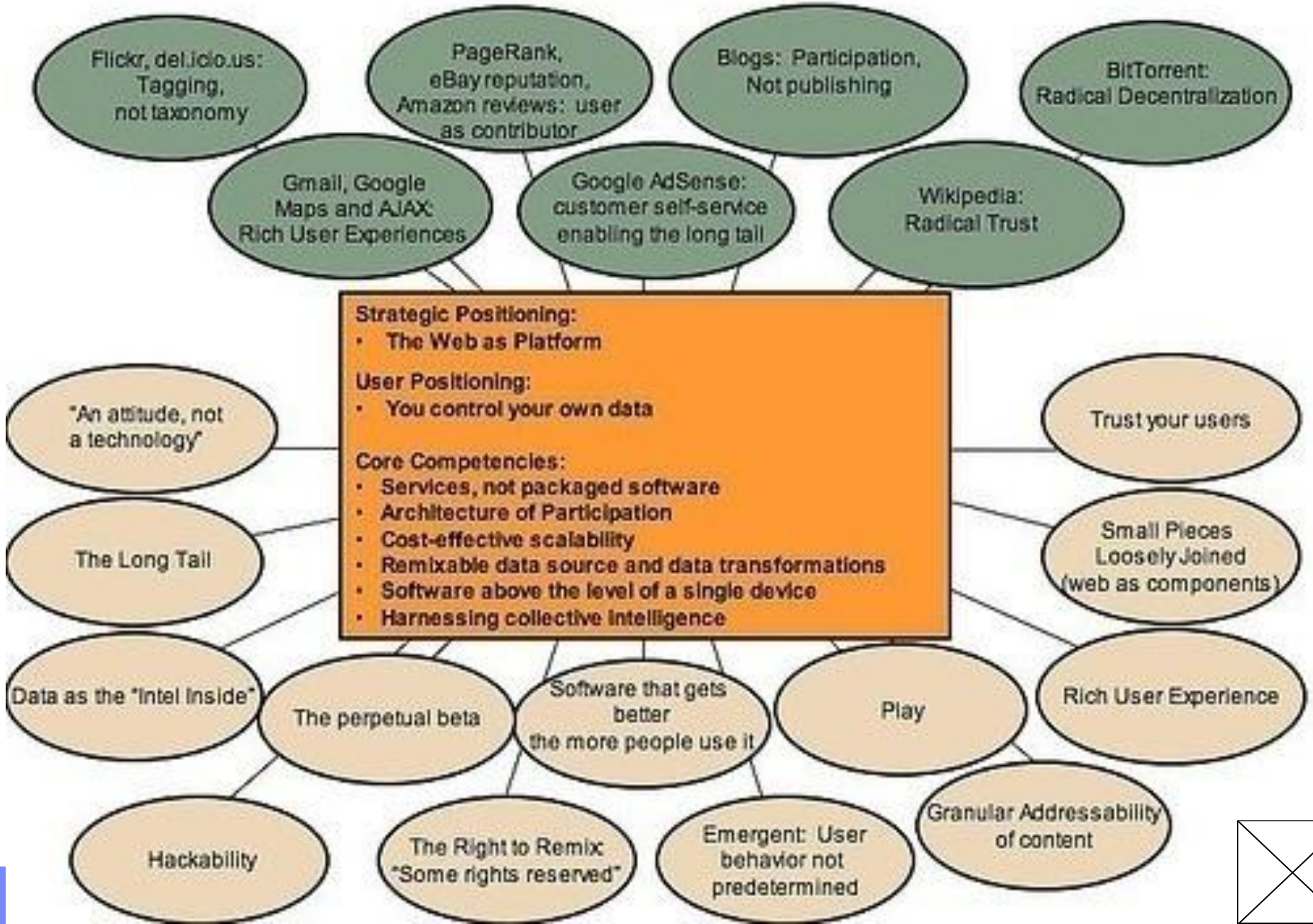


Web2.0

- **Web 2.0的关键概念和技术**
  - **使用REST来表示和访问服务**
  - **数据被编码成XML文档或者ATOM Feed（json）以用来交换数据**
  - **基于Ajax或者RIA的丰富用户体验**



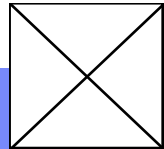
Web 2.0 Meme Map



Web2.0

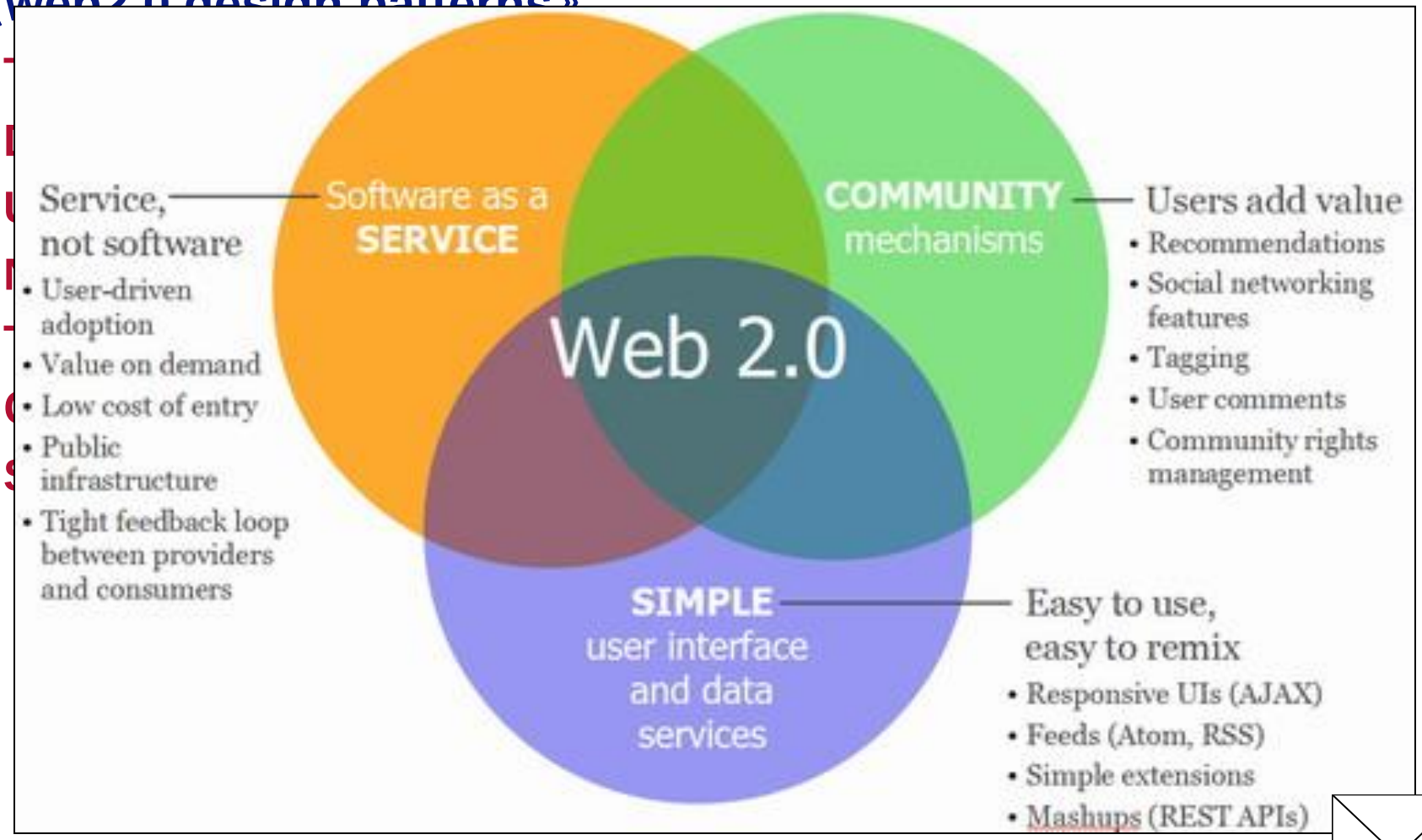
Web1.0（1993 ～ 2003）通过浏览器浏览网页		Web2.0（2003～ ）网页，以及许多通过web 分享的其它内容，更为互动，更象有应用功能而不仅仅是一个网页
读	模式	写与贡献
网页	主要内容单元	发表/ 记录的信息
静态	形态	动态
互联网浏览器	浏览方式	各类浏览器、RSS 阅读器、其它
Client Server	体系结构	Web Services
程序员	内容建立者	人人
初级的“滑稽”应用	应用领域	大量成熟应用

银行信息中心系统



Web2.0

- 《web2.0 design patterns》



Web2.0

- 应用
  - Blog
    - Weblo
    - 源于 w
      录，并
      和回响
    - BSP:网
    - BCP:网
      - 服务

我的电影 - Mozilla Firefox

文件(F) 编辑(E) 查看(V) 历史(S) 书签(B) 工具(T) 帮助(H)

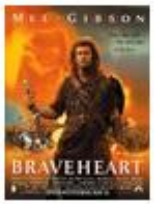
http://www.douban.com/people/1504535/movies

Windows Media | Windows | 免费 Hotmail | 自定义链接 | 中国XML论坛 - 『 D... | W3CHINA.ORG讨论区 ...

我想看

> 所有想看的电影 (1)

我看过



我的电影标签

侯孝賢 坚强 经典 浪漫 励志 亲情 青春 人情 人性 善良 生活 台湾电影 童年 往事 温暖 希望 信念 性情中人 真诚 执着 自然 自由

豆瓣猜你会喜欢 (更多推荐)

Fabuleux destin d'Amélie Poulain, Le x

Schindler's List x

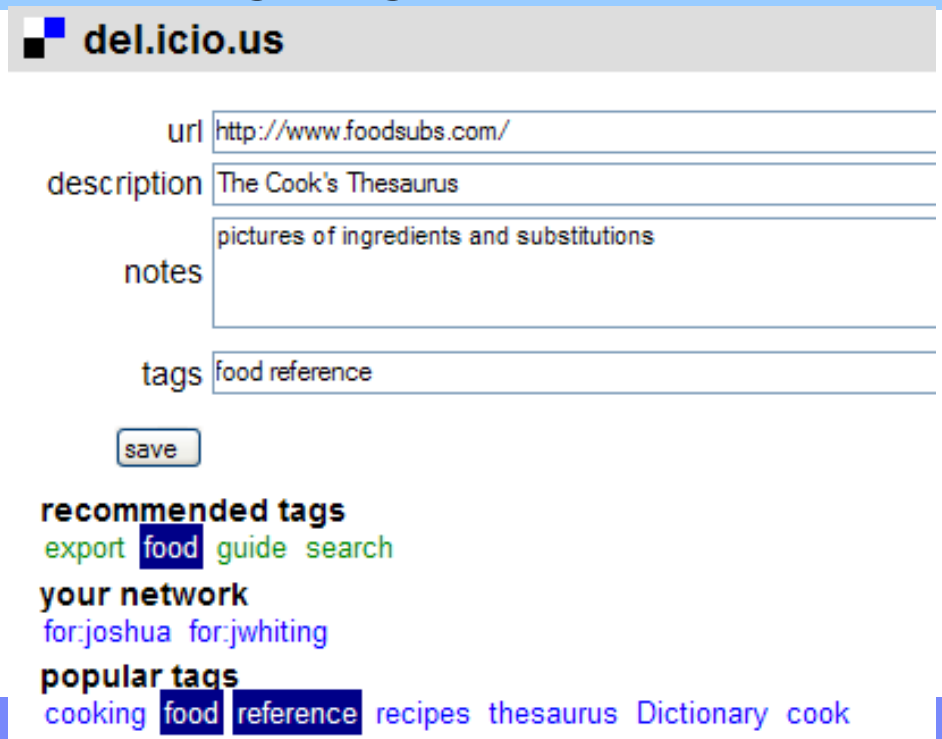
千と千尋の神隠し x

完成

Web2.0

Tags

A tag is simply a word you use to describe a bookmark. Unlike folders, you make up tags when you need them and you can use as many as you like. The result is a better way to organize your bookmarks and a great way to discover interesting things on the Web.




Second Life: Your World. Your Imagination. - Mozilla Firefox
http://www.secondlife.com/
Resident Login | Join
SECOND LIFE
Your World. Your Imagination.
WHAT IS SECOND LIFE? | SHOWCASE | BUSINESS & EDUCATION | DEVELOPERS | COMMUNITY | BLOG | SUPPORT
Search Second Life
JOIN NOW
MEMBERSHIP IS FREE
SECOND LIFE IS A 3D ONLINE DIGITAL WORLD IMAGINED, CREATED, & OWNED BY ITS RESIDENTS.
BUY & SELL LINDEN DOLLARS
OWN VIRTUAL LAND
REFER FRIENDS
VIEW INTERACTIVE MAP
UNDER 18? CHECK OUT TEEN SECOND LIFE
Total Residents: 5,351,104
Logged In Last 60 Days: 1,652,979
Online Now: 22,189
US$ Spent Last 24h: $1,559,060
LindeX Activity Last 24h: $232,590
SEE MORE economic statistics here!
Headlines read more news...
USA Today
Faithful build a Second Life for religion ...
Washington Post
French Politics in 3D on
second life the official guide
YOU MIGHT BE A LINDEN!
Work at Linden Lab
Where else can you help create a
Second Life Videos
Check out resident-created Second
正在从 www.secondlife.com 传送数据

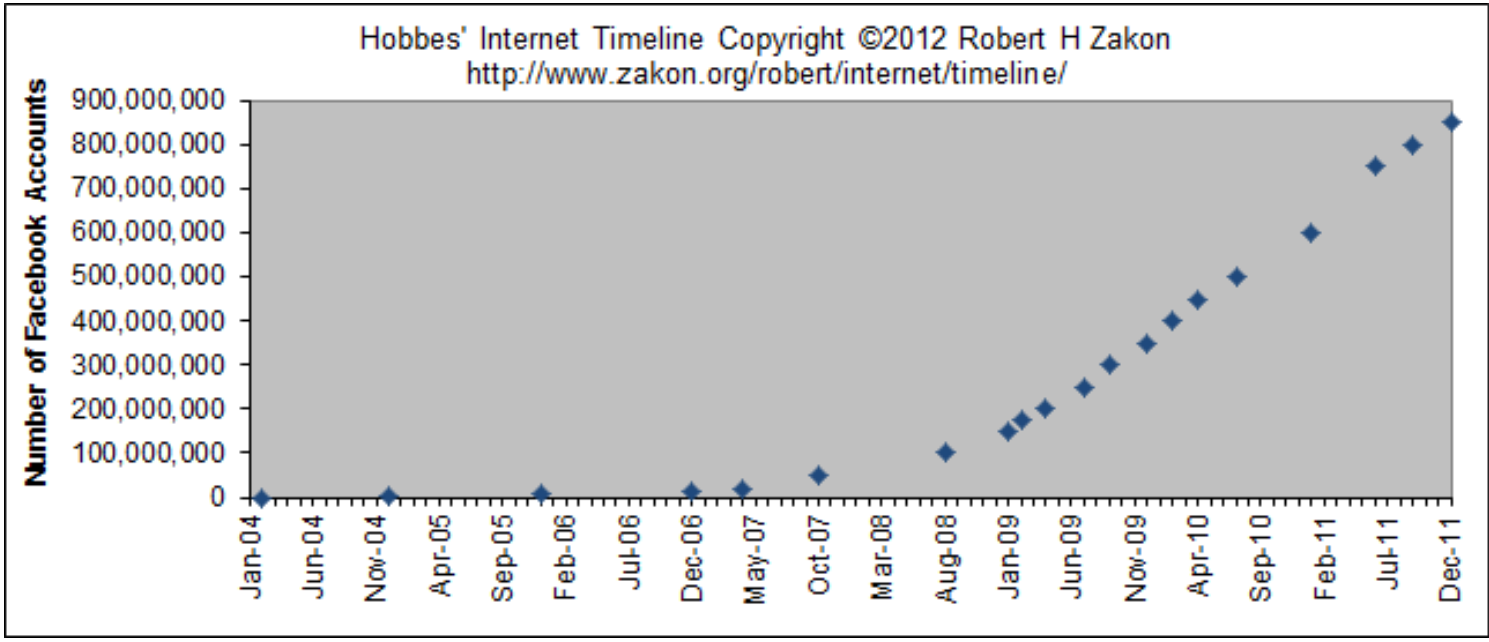
Web2.0

- 应用
  - SNS
    - Facebook
      - 发源于哈佛大学，是目前社会化网络和web2.0的风向标
      - 截止*2007*年9月26日，共有超过4500个*Facebook*应用出现
      - 网站功能
        - 墙（The Wall）
        - 礼物（Gift）
        - 市场（Marketplace）
        - 捅（Pokes）
        - 状态（Status）
        - 活动（Events）
        - 开放平台上的应用（Application）

*Facebook*创始人兼CEO Mark Zuckerberg

Web2.0

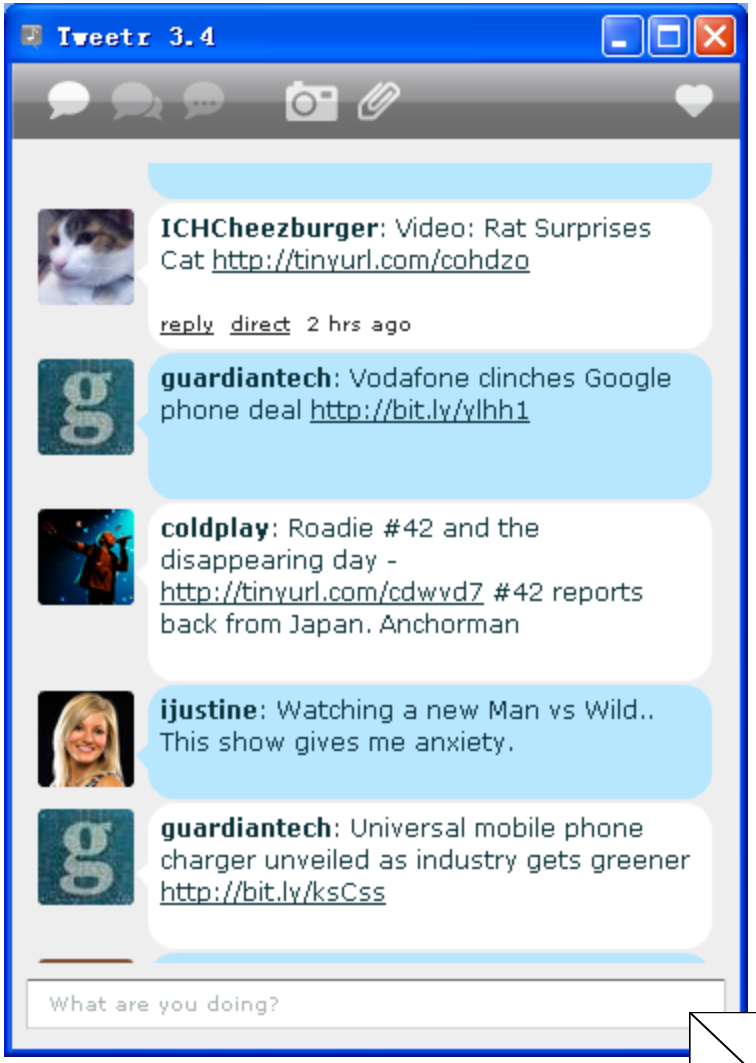
- 应用
  - SNS
    - Facebook



Web2.0

- 应用
  - SNS
    - 推客twitter

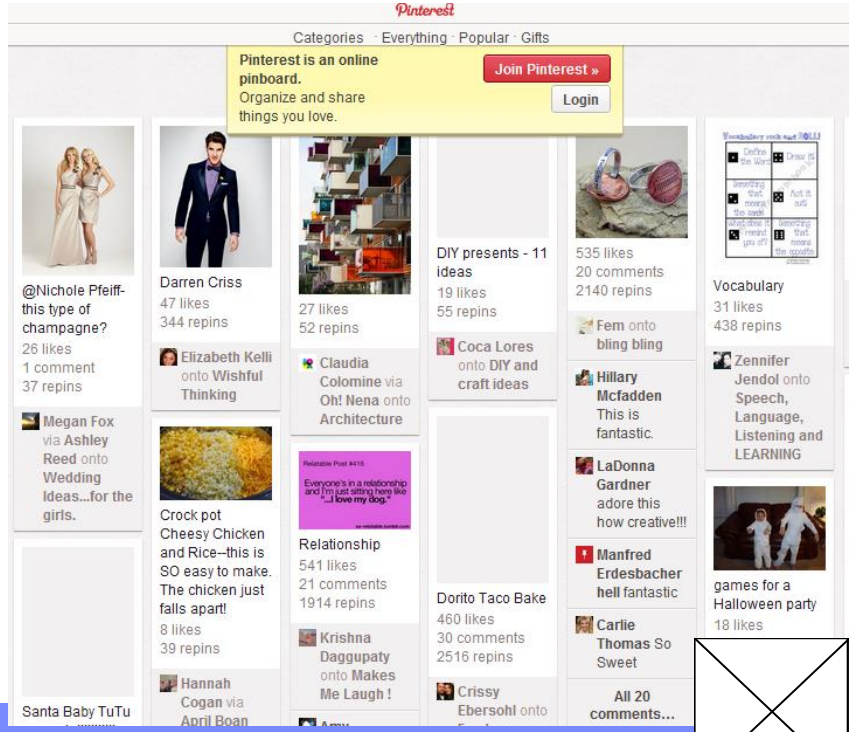
一个web2.0式的聊天/短信互动平台,一种微型博客服务，你可以用它通过手机短信(SMS short message service)、即时通讯(IM instant message)或网上在线(web)中的任何一种方式向自己的账户页面发布信息告诉别人此时此刻你在干什么，在想什么。



NVE的应用

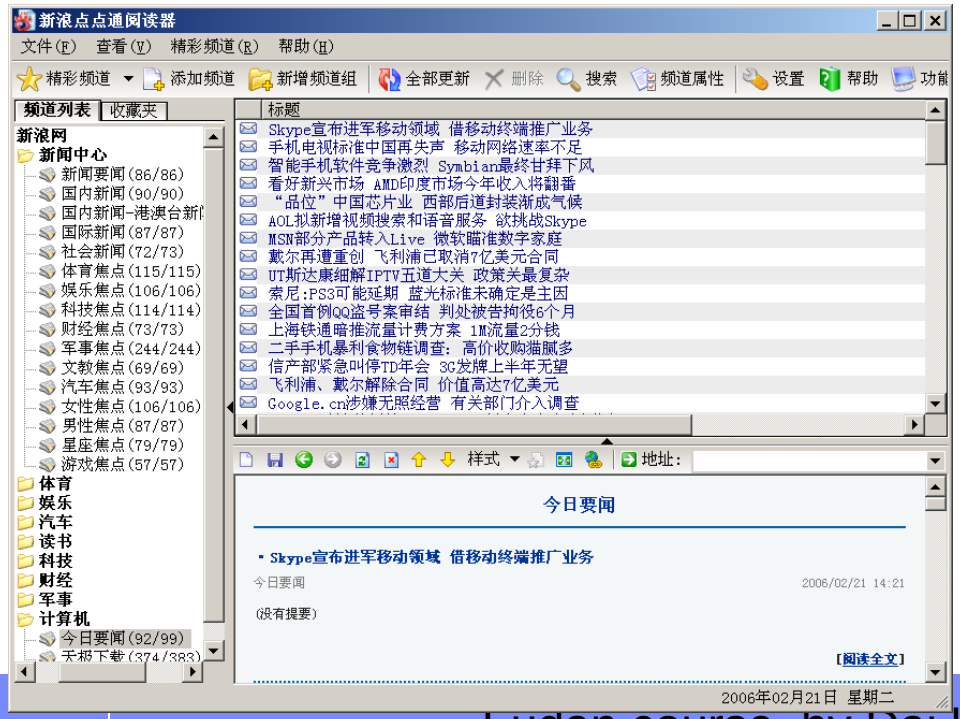
- 应用
  - SNS

Facebook
Twitter
Linkedin
Myspace
Google+
tumblr
Pinterest



Web2.0

- 应用
  - RSS
    - RSS是一种用于共享新闻和其他Web内容的数据交换规范，起源于网景通讯公司的推"Push"技术
    - Rich Site Summary（丰富站点摘要）



RSS阅读器比较

UNIX pipe of the internet

Web2.0

- 应用
  - RSS

Web2.0

- 应用
  - Wiki

Create account Log in

Article Talk Read Edit View history Search

Web 2.0

From Wikipedia, the free encyclopedia
(Redirected from Web2.0)

Web 2.0 is a concept that takes the network as a platform for information sharing, interoperability, user-centered design, and collaboration on the World Wide Web. A Web 2.0 site allows users to interact and collaborate with each other in a social media dialogue as creators (prosumers) of user-generated content in a virtual community, in contrast to websites where users (consumers) are limited to the passive viewing of content that was created for them. Examples of Web 2.0 include social networking sites, blogs, wikis, video sharing sites, hosted services, web applications, mashups and folksonomies.

The term is closely associated with Tim O'Reilly because of the O'Reilly Media Web 2.0 conference in late 2004.[1][2] Although the term suggests a new version of the World Wide Web, it does not refer to an

A tag cloud (a typical Web 2.0 phenomenon in itself) presenting Web 2.0 themes. An intera version is available here.

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact Wikipedia

Toolbox

Print/export

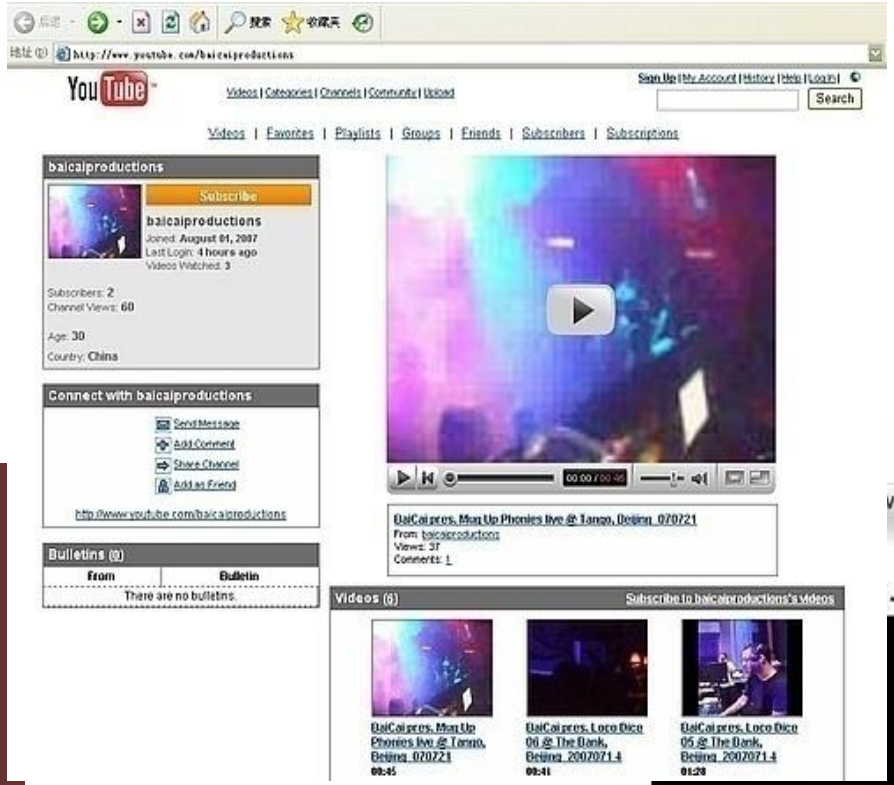
Languages

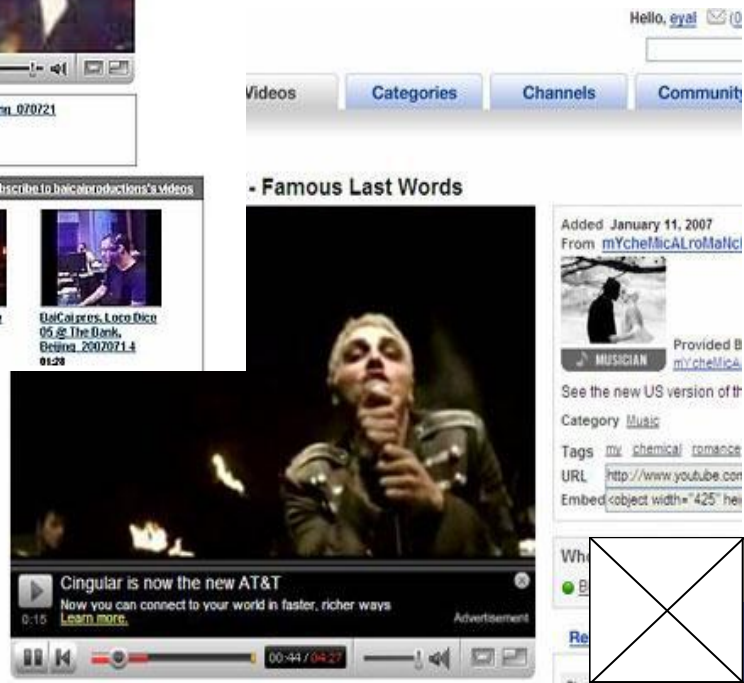
Web2.0

- 应用
  - Podcasting：
    - 个人视频/声频的发布/定阅

- **Youtube.com founder**
- **Google bought YouTube: $1.65 billion in a stock-for-stock transaction**

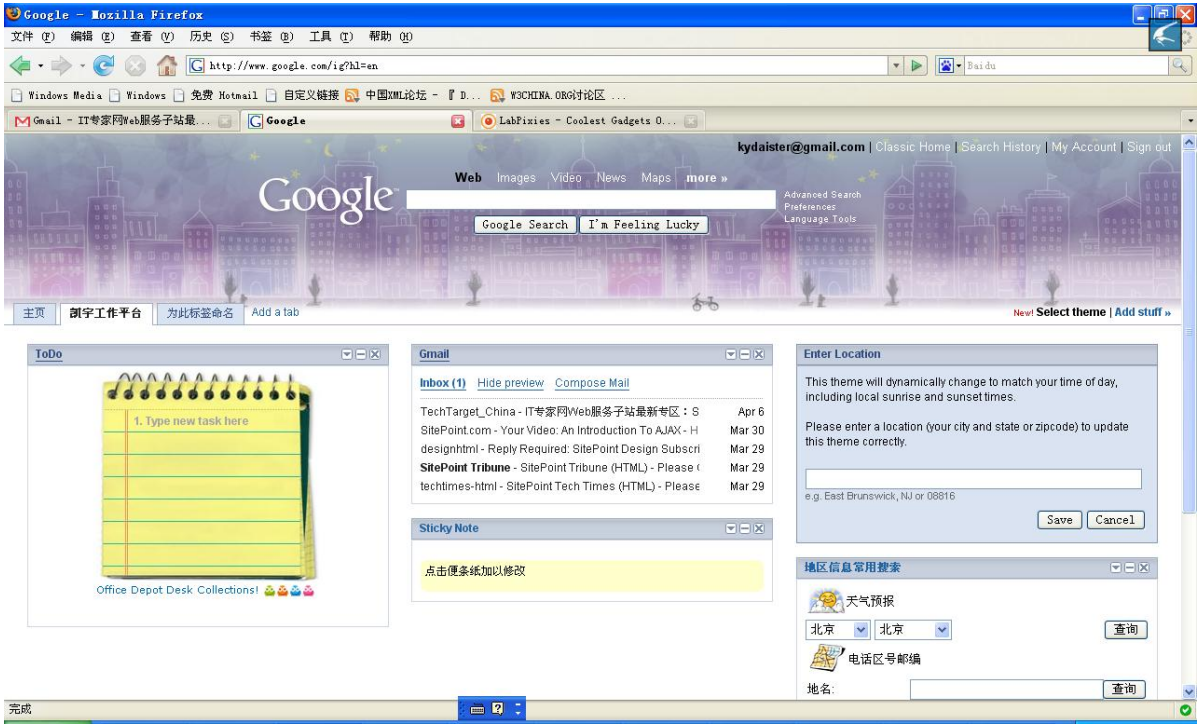
陈士骏 (Steve Chen)
Chad Hurley





Web2.0

- 应用
  - 个性化门户

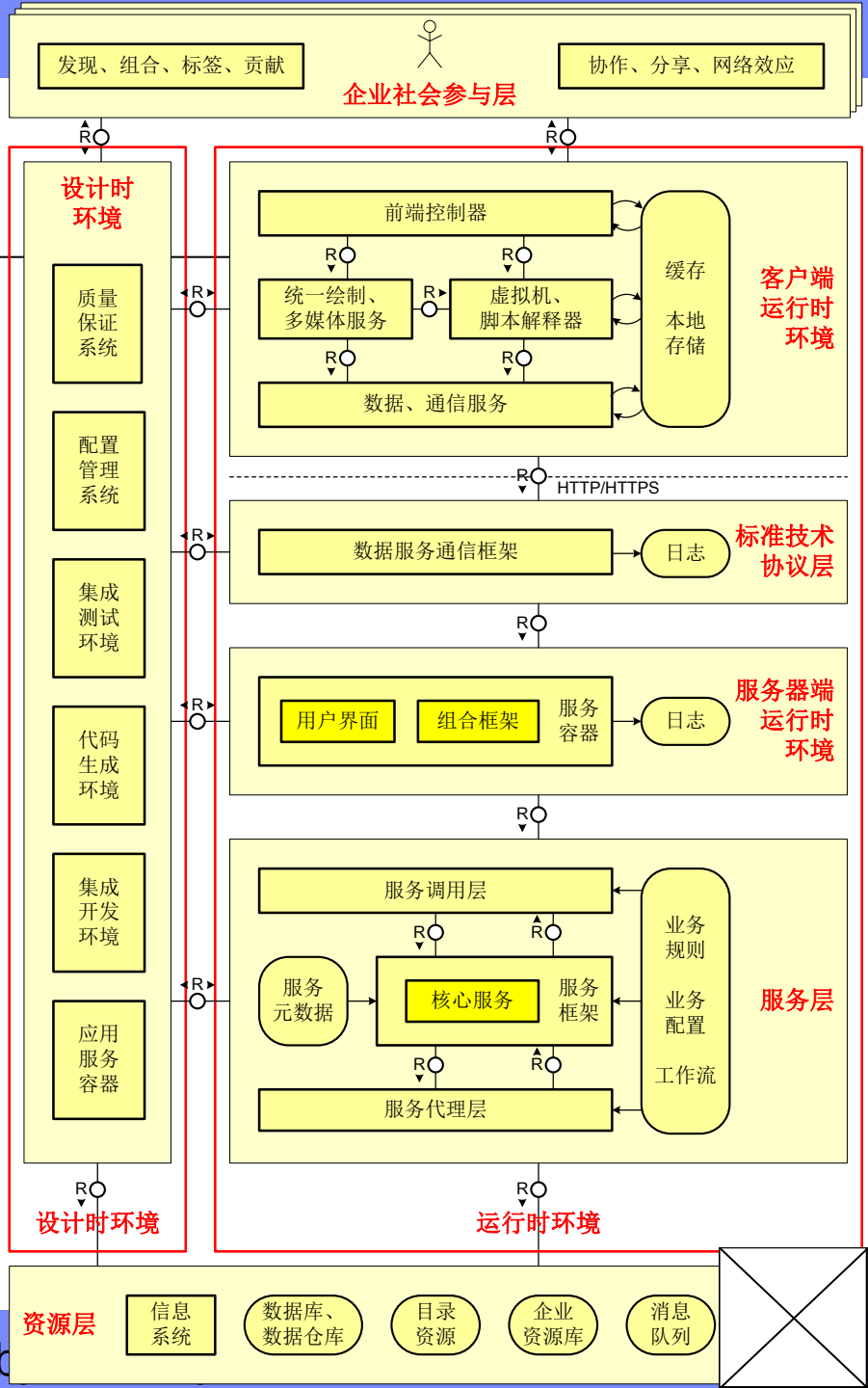


Web2.0 + SOA

- 面向Web的组合环境愿景视图

Web2.0 + SOA

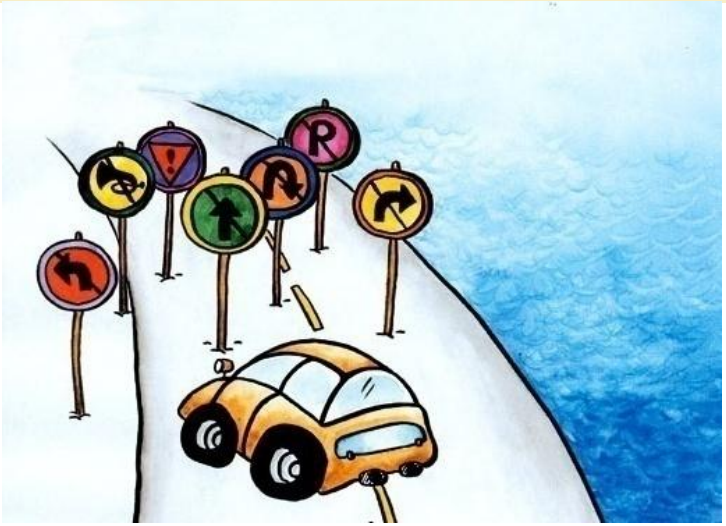
- 面向Web的组合环境参考架构



Rich Internet Application技术

- **Web的不足**

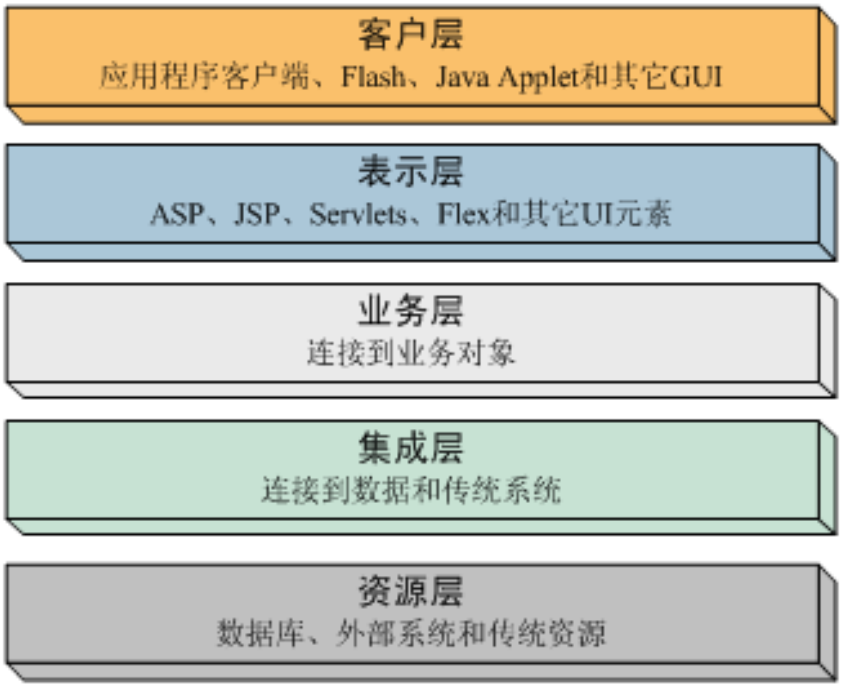
  - 现在的**Web**应用程序对完成复杂应用方面却始终跟不上步伐
  - **Web**模型是基于页面的模型，缺少客户端智能机制
  - 几乎无法完成复杂的用户交互（如传统的**C/S**应用程序和桌面应用程序中的用户交互）



Business 2.0 magazine (October 2004) selected Rich Internet Applications as one of the ten technologies to watch in 2005

RIA的应用程序模型

- **RIA中的 Rich Client（丰富客户端）提供可承载已编译客户端应用程序（以文件形式，用HTTP传递）的运行环境，客户端应用程序使用异步客户/服务器架构连接现有的后端应用服务器**

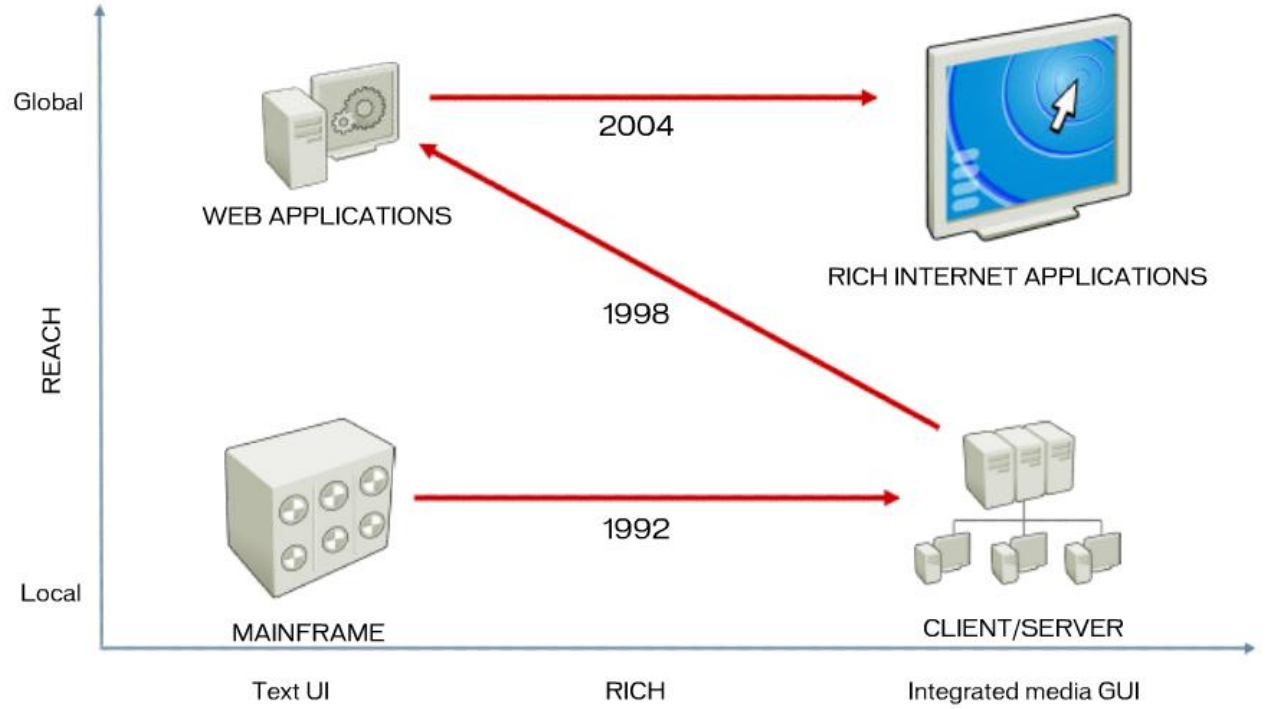


Rich Internet Application技术

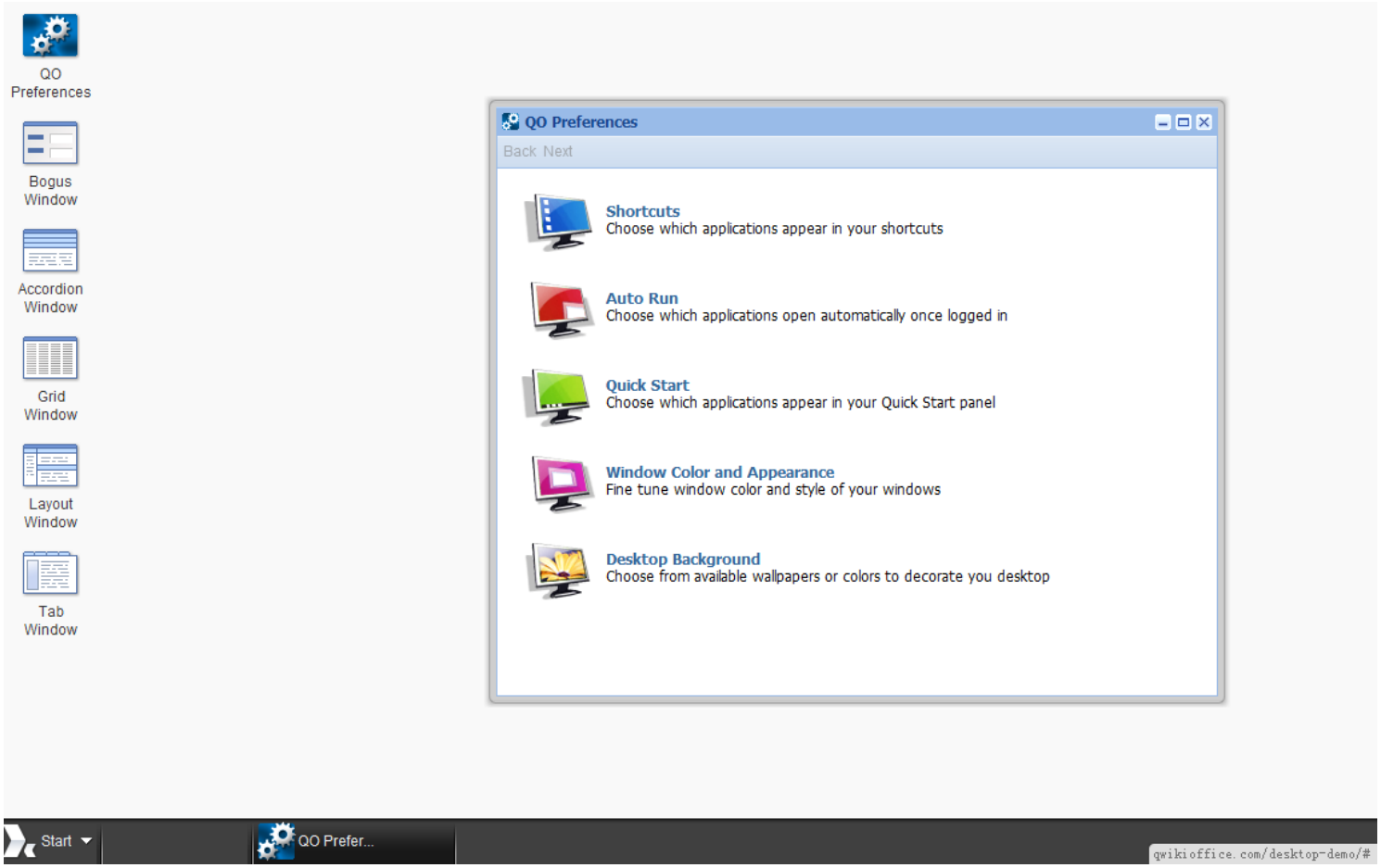
- **Rich在哪里？**
  - *数据模型的丰富*
    - 用户界面可以显示和操作更为复杂的嵌入在客户端的数据模型
  - *用户界面的丰富*
    - 提供了灵活多样的界面控制元素，这些控制元素可以很好的与数据模型相结合
  - *最好的通讯模式*
    - 无刷新页面之下提供快捷的界面响应时间
    - 双向互动声音和图像。

Rich Internet Application技术

- **Rich Internet Applications横空出世**
  - 结合了桌面应用程序的反应快、交互性强的优点
  - Web应用程序的传播范围广及容易传播，低成本部署的特性
  - RIA简化并改进了Web应用程序的用户交互。这样，用户开发的应用程序可以提供更丰富、更具有交互性和响应性的用户体验



RIA例子



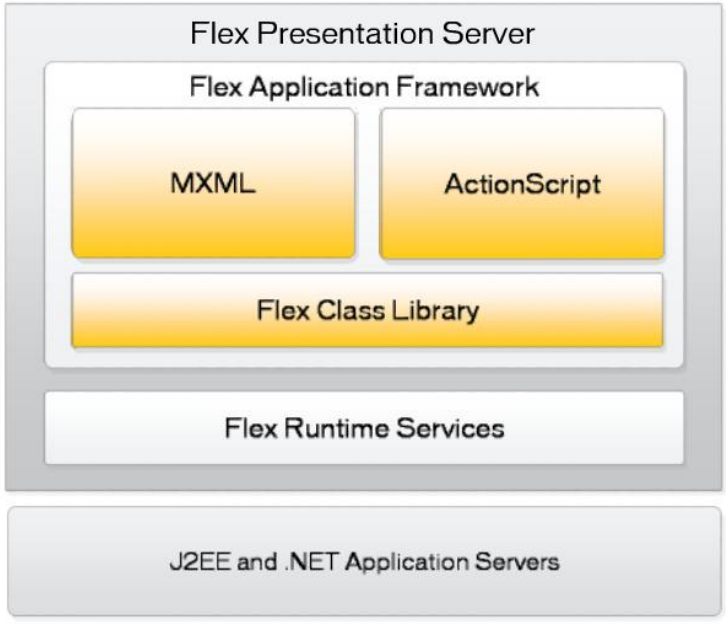
http://qwikioffice.com/desktop-demo/

Rich Internet Application技术

- **Java WebStart– sun**
  - 从版本 1.3 开始，JRE 就包含了 Java Webstart
  - 简单但功能强大且灵活的将应用程序部署到任何平台的方法
  - Webstart 允许应用程序的用户从他们的浏览器、电子邮件或桌面启动和管理应用程序（只需在网页上点击一个超级链接就能运行一个Java桌面应用 ）
  - 直接从服务端发布到客户端
  - 检查用户是否在运行最新版本的应用程序
  - http://java.sun.com/products/javawebstart/index.jsp

Rich Internet Application技术

- **FLEX --Macromedia**
  - 将MXML(the Macromedia Flex Markup Language)文件，编译成SWF文件，然后显示在浏览器中,并利用Web Service技术和服务器通信
  - Flex是为满足希望开发 RIA的企业级程序员的需求而推出的表示服务器和应用程序框架，它可以运行于J2EE和.NET平台
  - **http://www.macromedia.com/software/flex/**



macromedia FLEX

Rich Internet Application技术

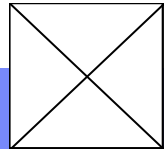
- **Jdnc (java desktop behind) – sun**
  - **https://jdnc.dev.java.net/**

一个基于Novell公司的SuSE Linux的桌面Linux操作系统

Rich Internet Application技术

- ***XUL***
  - XML用户接口语言，（XML User interface Language）
  - 一个基于XML的用户界面语言，它来自于Mozilla的开放源码项目，XUL描述引擎都非常小（100K以下），
  - XUL是一种用于交换数据的标准，它主要用于描述用户接口，具有独立于平台的特点，十分灵活，可以和XML等现在流行的语言相兼容
  - XUL最大的优点在于它与Gecko引擎的集成（打开了通向大量Web标准的大门）

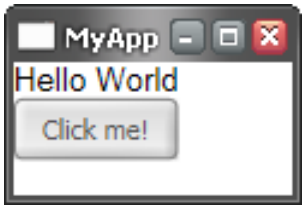
- **Bindows**
  - Bindow 是用Javascript和DHTML开发的Web窗体框架。Javascript用于客户端界面的显示和处理，XMLHTTP用于客户端与服务器的信息传输
  - Bindows的一个主要的缺点是它采用一次全部载入的方式来实现脚本库
  - 没有考虑到非IE的浏览器，限制了Bindows的流行

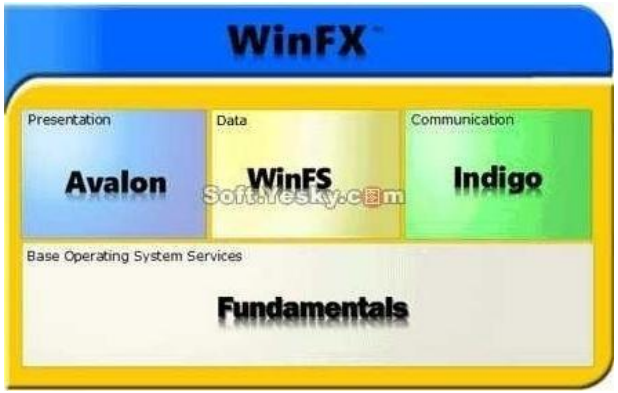


Rich Internet Application技术

- **XAML and Microsoft SilverLight**
  - XAML的全称是eXtensible Application Markup Language，由Avalon延伸而来，用于像HTML构建WEB页面一样构建应用程序的界面，XAML基于XML

```
<FlowPanel xmlns="http://schemas.microsoft.com/2003/xaml">
   <Text>Hello World</Text>
   <Button>Click me!</Button>
</FlowPanel>
```

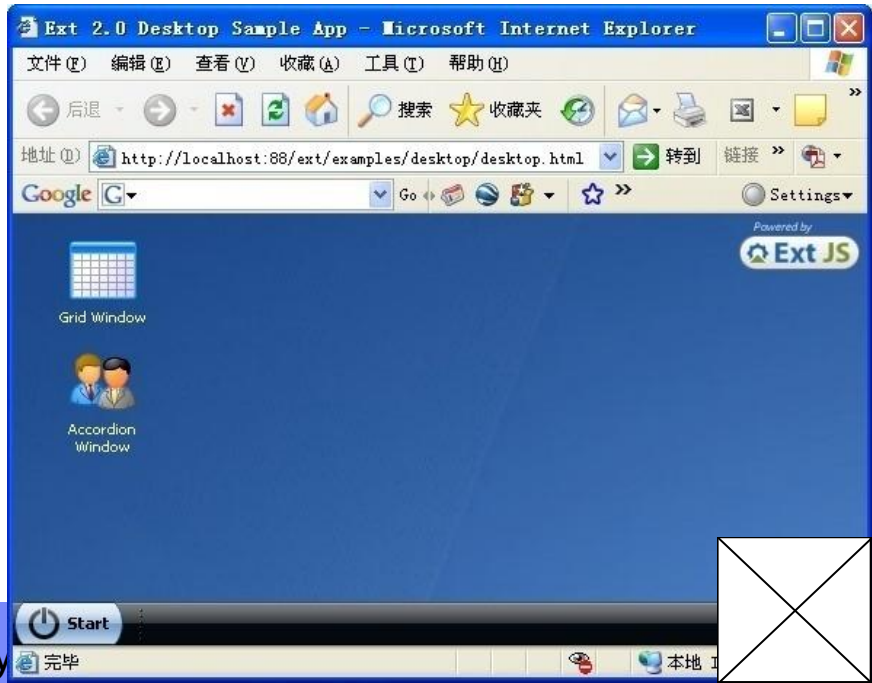




Rich Internet Application技术

- 开源的RIA开发环境Laszlo
  - 编写名为LZX的描述语言（其中整合了XML和JavaScript），运行在J2EE应用服务器上的Laszlo平台会将其编译成FLASH文件并传输给客户端展示
  - http://www.openlaszlo.org

- Ext.
  - JS库框架
  - 内建卓越的Ajax支持
  - http://extjs.com/downloads



Demo (in IIS)
http://localhost:88/ext/examples/samples.html
http://localhost:88/ext/examples/portal/portal.html

Rich Internet Application技术

- Native Client **of Google**

Native Client is an open-source technology for running native code in web applications, with the goal of maintaining the browser neutrality, OS portability and safety that people expect from web appsWe believe that Native Client technology will help web developers to create richer and more dynamic browser-based applications.

Quake in the browse

http://code.google.com/p/nativeclient/

Ajax简介

- **Ajax: A New Approach to Web Applications**
  - **最早提出者Jesse James Garrett**
  - **http://www.adaptivepath.com/publications/essays/archives/000385.php**

adaptive path

blog | about us | services | events | publications | products | contact | home

publications

reports | latest essay | essay archives | newsletter | reading list

Ajax: A New Approach to Web Applications

Jesse James Garrett is President and a founder of Adaptive Path. He is the author of the widely-referenced book The Elements of User Experience. Jesse's other

by Jesse James Garrett

February 18, 2005

If anything about current interaction design can be called "glamorous," it's creating Web applications. After all, when was the last time you heard someone rave about the interaction design of a product that wasn't on the Web? (Okay, besides the iPod.) All the cool, innovative new projects are online.

Despite this, Web interaction designers can't help but feel a little envious of our colleagues who create desktop software. Desktop applications have a richness and responsiveness that has seemed out of reach on the Web. The same simplicity that enabled the Web's rapid proliferation also creates a gap between the experiences we can provide and the experiences users can get from a desktop application.

That gap is closing. Take a look at Google Suggest. Watch the way the suggested terms update as you type, almost instantly. Now look at Google Maps.

Recent Essays

Sarah Nelson Interviews Scott Berkun at MX San Francisco
February 22, 2007

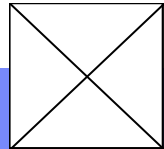
Nine Adaptive Pathers Share Their Resolutions for 2007
January 4, 2007

Interview with Tim Brown, CEO of IDEO
January 3, 2007

Tagging vs. Cataloging: What It's All About
November 30, 2006

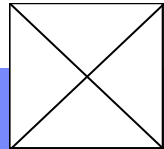
Organizing Your Global Corporate Intranet
November 10, 2006

Essay Archives »

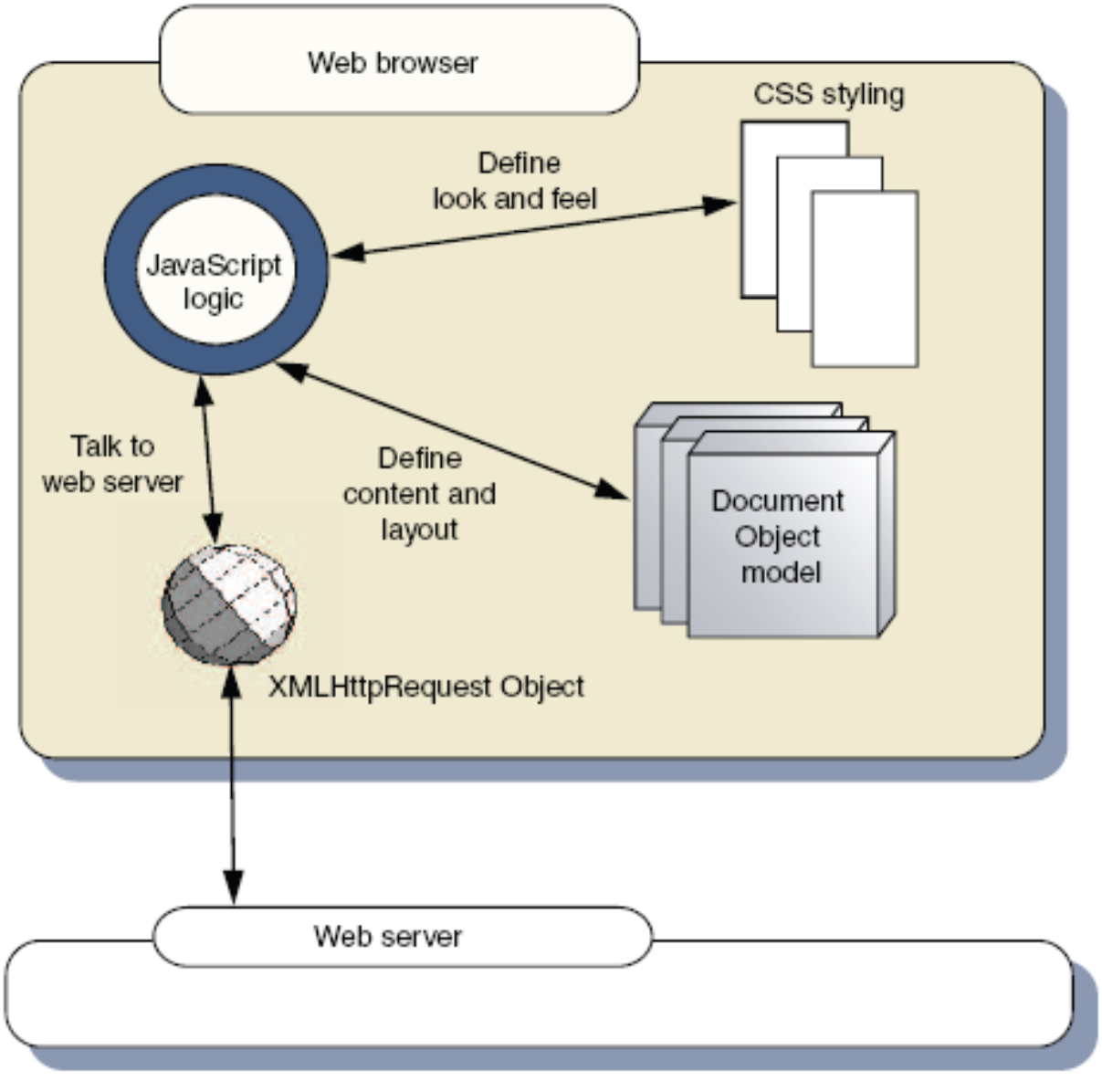


Ajax简介

- Web2.0的核心技术
- Ajax是Asynchronous JavaScript and XML的缩写
- 有确定需要从服务器读取新数据时再由Ajax引擎代为向服务器提交请求。
  - 使用XHTML和CSS标准化呈现
  - 使用DOM实现动态显示和交互
  - 使用XML和XSLT进行数据交换与处理
  - 使用XMLHttpRequest进行异步数据读取，向XMLHttpRequest注册一个回调函数
  - 最后用JavaScript整合以上技术

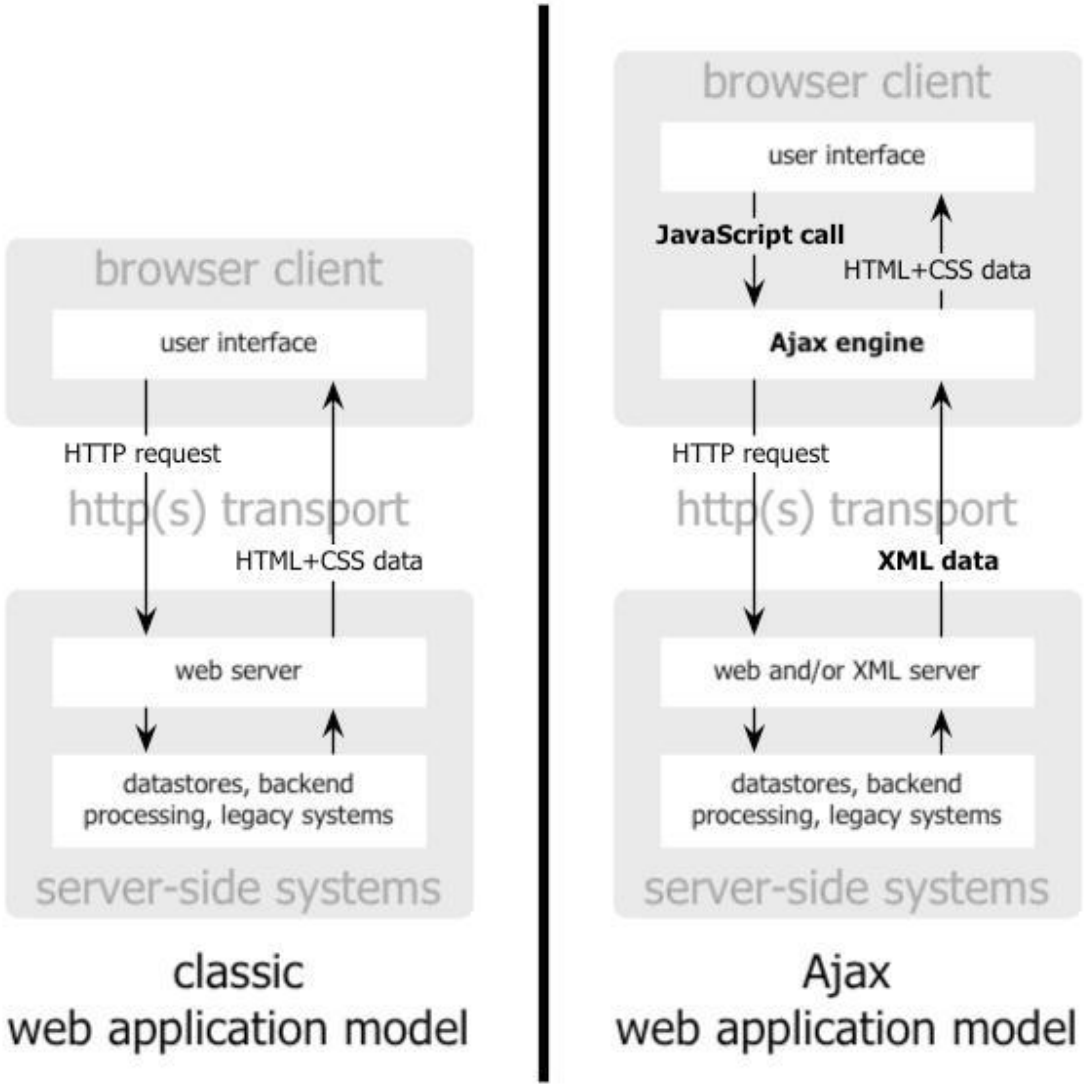


Ajax简介


Web browser
CSS styling
Define
look and feel
JavaScript
logic
Talk to
web server
Define
content and
layout
Document
Object
model
XMLHttpRequest Object
Web server

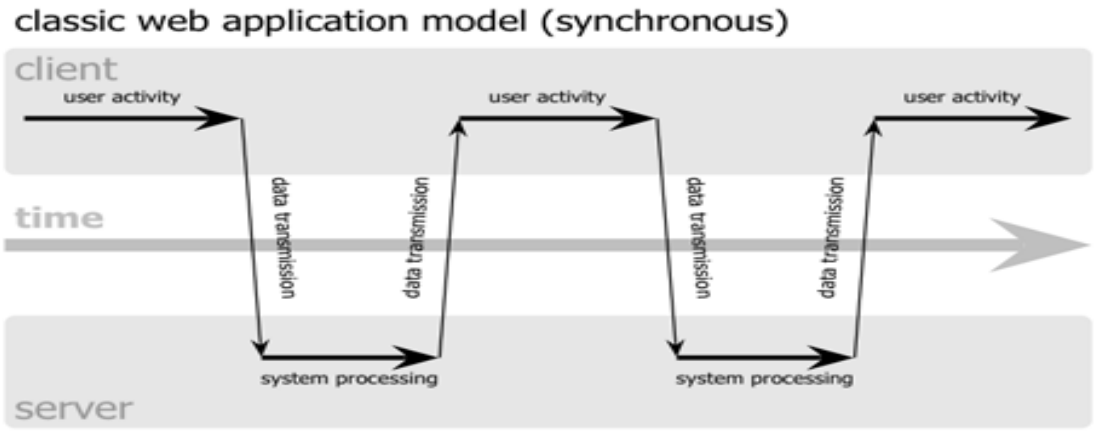
Ajax简介

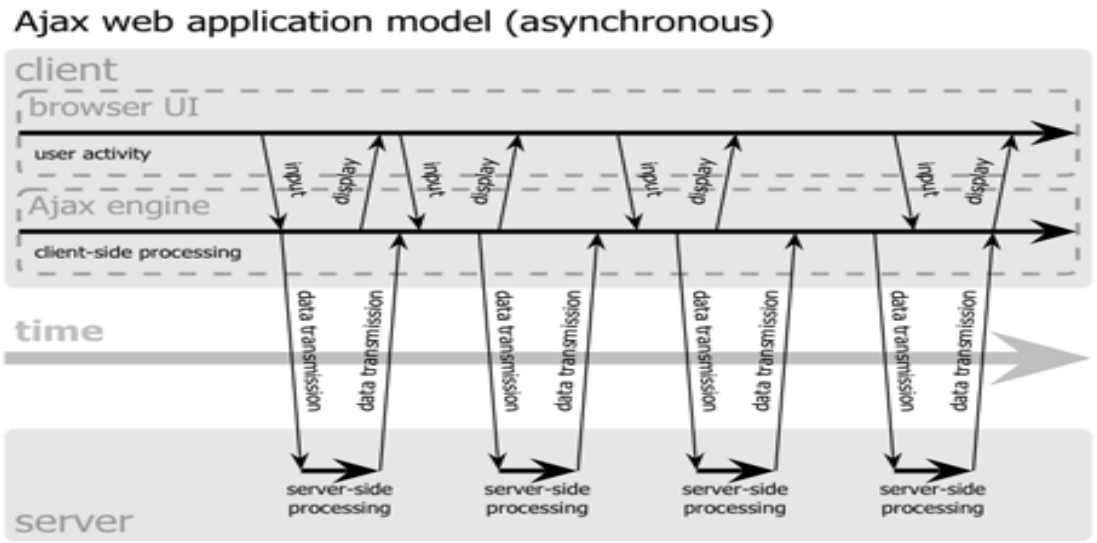
- JavaScript编写的Ajax引擎允许用户与页面的交互和页面与服务器的交互异步的



Ajax简介

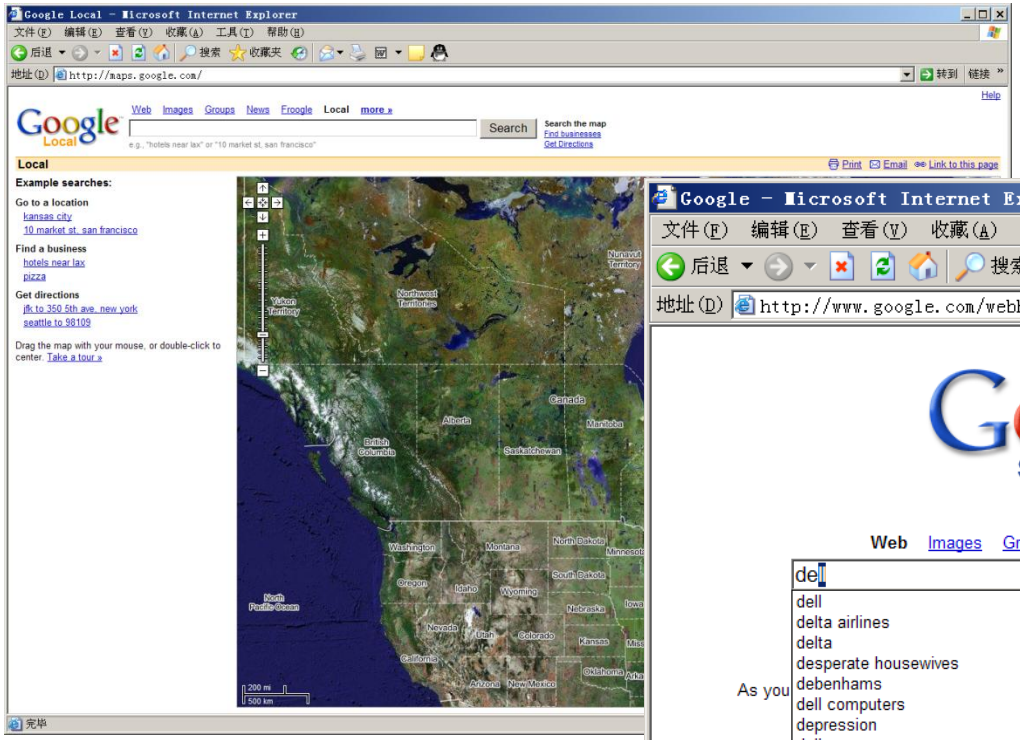
- 异步调用模型

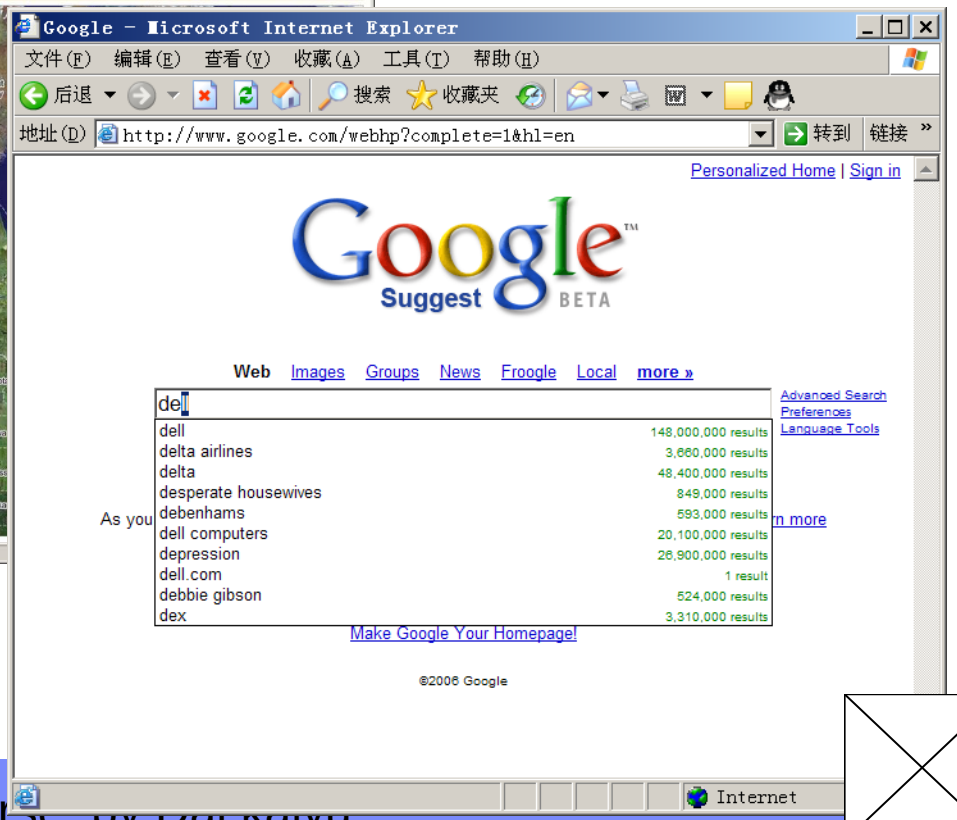




Ajax简介

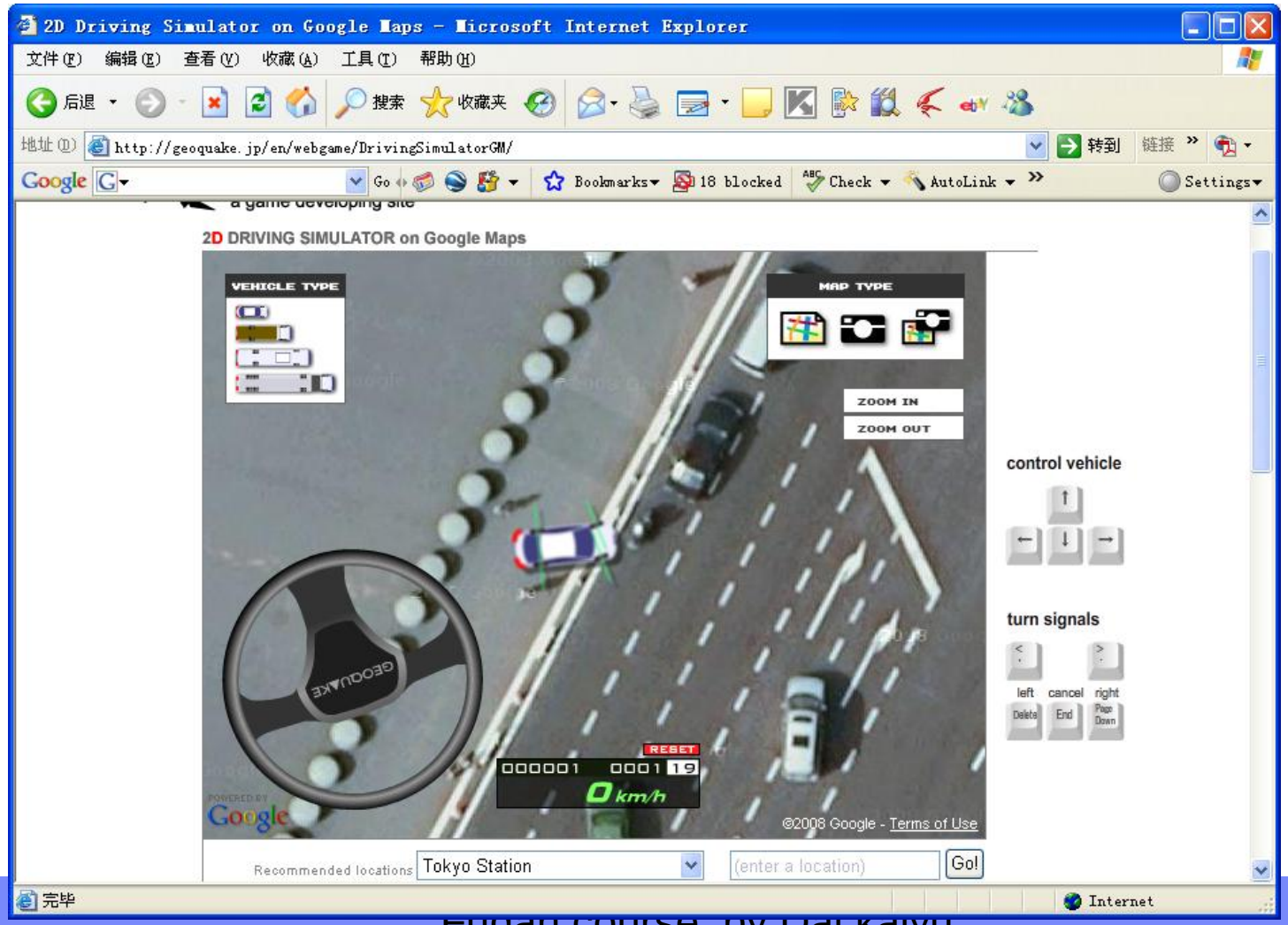
应用：**Orkut，Gmail，Google Groups，Google Suggest，以及Google Maps**





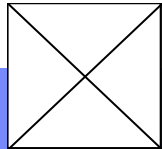
基于google map的游戏

http://geoquake.jp/en/webgame/DrivingSimulatorGM/



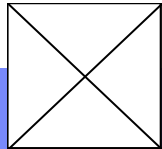
AJAX

- DHTML plus Asynchronous communication capability through XMLHttpRequest
- Pros
  - Most viable RIA technology so far
  - Tremendous industry momentum
  - Several toolkits and frameworks are emerging
  - No need to download code & no plug-in required
- Cons
  - Still browser incompatibility
  - JavaScript is hard to maintain and debug



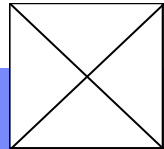
Key Aspects of Google Maps

- A user can drag the entire map by using the mouse
  - Instead of clicking on a button or something
- The action that triggers the download of new map data is not a specific click on a link but a moving the map around
- Behind the scene - AJAX is used
  - The map data is requested and downloaded asynchronously in the background
- Other parts of the page remains the same
  - No loss of operational context



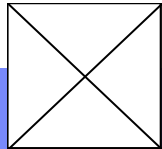
Usage cases for AJAX

- **Real-time server-side input form data validation**
  - User IDs, serial numbers, postal codes
  - Removes the need to have validation logic at both client side for user responsiveness and at server side for security and other reasons
- **Auto-completion**
  - Email address, name, or city name may be auto-completed as the user types
- **Advanced GUI widgets and controls**
  - Controls such as tree controls, menus, and progress bars may be provided that do not require page refreshes



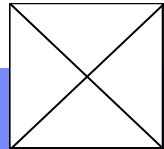
Usage cases for AJAX

- **Refreshing data**
  - HTML pages may poll data from a server for up-to-date data such as scores, stock quotes, weather, or application-specific data
- **Simulating server side notification**
  - An HTML page may simulate a server-side notification by polling the server in the background



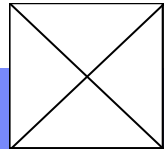
Server-Side AJAX Request processing

- **Server programming model remains the same**
  - **It receives standard HTTP GETs/POSTs**
  - **Can use Servlet, JSP, JSF, …**
- **With minor constraints**
  - **More frequent and finer-grained requests from client**
    - **Response content type can be**
      - **text/xml**
      - **text/plain**
      - **text/json**
      - **text/javascript**



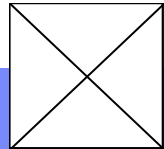
XMLHttpRequest Methods

- **open(“HTTP method”, “URL”, syn/asyn)**
  - Assigns HTTP method, destination URL, mode
- **send(content)**
  - Sends request including string or DOM object data
- **abort()**
  - Terminates current request
- **getAllResponseHeaders()**
  - Returns headers (labels + values) as a string
- **getResponseHeader(“header”)**
  - Returns value of a given header
- **setRequestHeader(“label”,”value”)**
  - Sets Request Headers before sending



XMLHttpRequest Properties

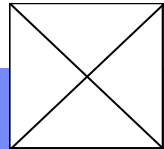
- **onreadystatechange**
  - Set with an JavaScript event handler that fires at each state change
- **readyState – current status of request(from 《Professional Ajax》)**
  - **0 :(Uninitialized) The object has been created but the open() method hasn't been called.**
  - **1 :(Loading) The open() method has been called but the request hasn't been sent.**
  - **2 :(Loaded) The request has been sent .**
  - **3 :(Interactive) A partial response has been received.**
  - **4 :(Completed) All data has been received and the connection has been closed .**
- **status**
  - HTTP Status returned from server: 200 = OK



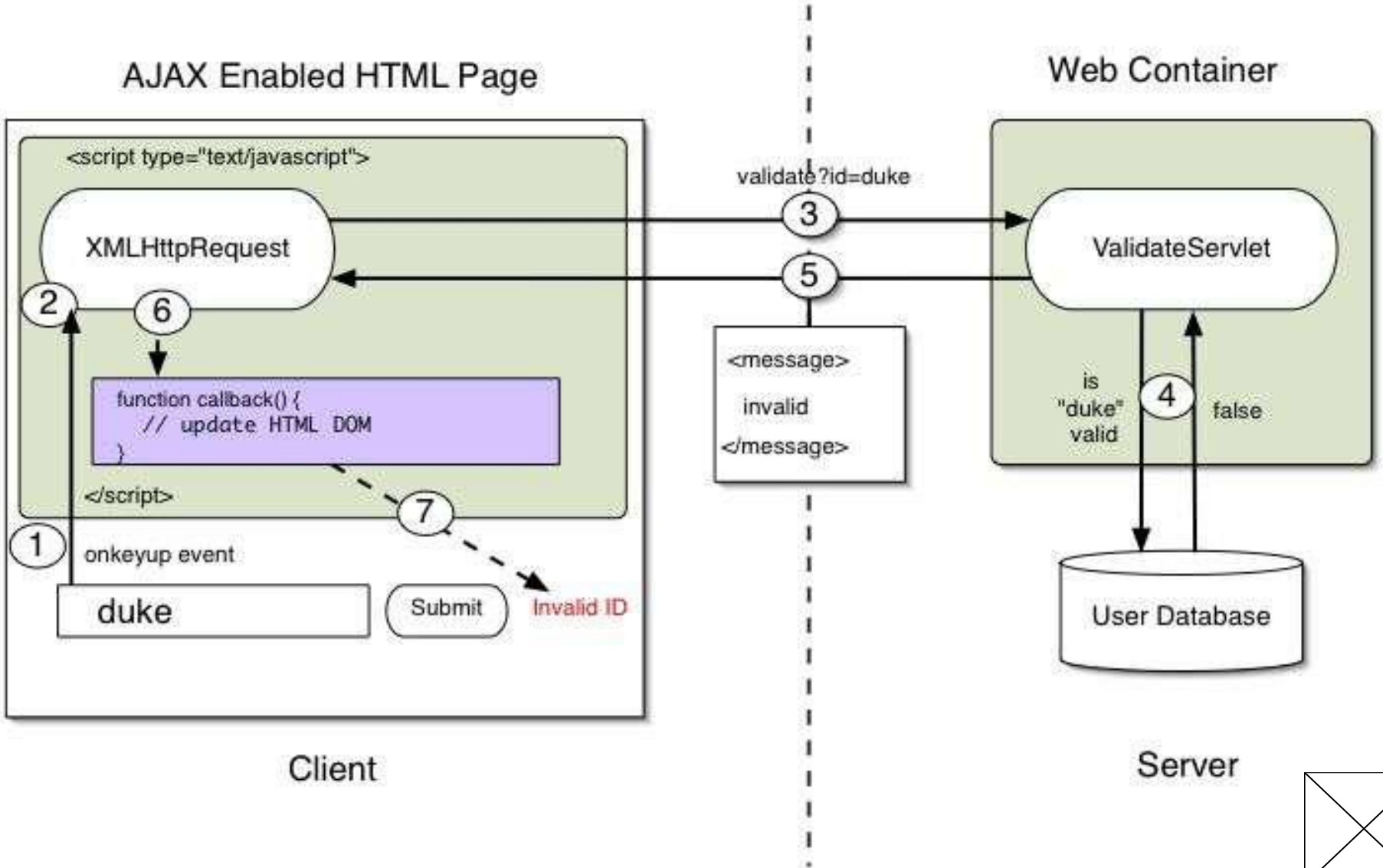
XMLHttpRequest Properties

- **responseText**
  - String version of data returned from the server
- **responseXML**
  - XML document of data returned from the server
- **statusText**
  - Status text returned from server

属性	描述
onreadystatechange	状态改变的事件触发器
readyState	对象状态(integer): 0 = 未初始化 1 = 读取中 2 = 已读取 3 = 交互中 4 = 完成
responseText	服务器进程返回数据的文本版本
responseXML	服务器进程返回数据的兼容 DOM 的 XML 文档对象
status	服务器返回的状态码，如：404 = "文件未找到"、200 ="成功"
statusText	服务器返回的状态文本信息

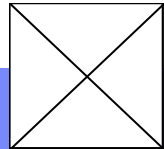


Data Validation Example



Steps of AJAX Operation

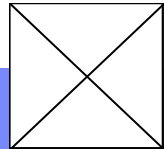
1. A client event occurs
2. An XMLHttpRequest object is created
3. The XMLHttpRequest object is configured
4. The XMLHttpRequest object makes an async. request
5. The ValidateServlet returns an XML document containing the result
6. The XMLHttpRequest object calls the callback() function and processes the result
7. The HTML DOM is updated



1. A Client event occurs

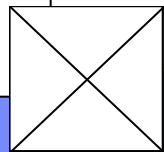
- A JavaScript function is called as the result of an event
- Example: validateUserId() JavaScript function is mapped as a event handler to a onkeyup event on input form field whose id is set to “userid”

```
<input type="text"
       size="20"
        id="userid"
       name="id"
    onkeyup="validateUserId();">
```



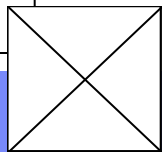
2. An XMLHttpRequest object is created

```
var req;
function initRequest() {
    if (window.XMLHttpRequest) {
        req = new XMLHttpRequest();
    } else if (window.ActiveXObject) {
        isIE = true;
        req = new ActiveXObject("Microsoft.XMLHTTP");
    }
}

function validateUserId() {
    initRequest();
    req.onreadystatechange = processRequest;
    if (!target) target = document.getElementById("userid");
    var url = "validate?id=" + escape(target.value);
    req.open("GET", url, true);
    req.send(null);
}
```



3. An XMLHttpRequest object is configured with a callback function

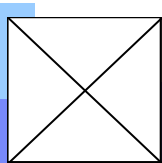
```
var req;
function initRequest() {
    if (window.XMLHttpRequest) {
        req = new XMLHttpRequest();
    } else if (window.ActiveXObject) {
        isIE = true;
        req = new ActiveXObject("Microsoft.XMLHTTP");
    }
}

function validateUserId() {
    initRequest();
    req.onreadystatechange = processRequest; // callback function
    if (!target) target = document.getElementById("userid");
    var url = "validate?id=" + escape(target.value);
    req.open("GET", url, true);
    req.send(null);
}
```



4. XMLHttpRequest object makes an async. request

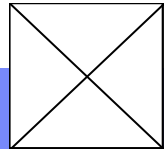
```
function initRequest() {
    if (window.XMLHttpRequest) {
        req = new XMLHttpRequest();
    } else if (window.ActiveXObject) {
        isIE = true;
        req = new ActiveXObject("Microsoft.XMLHTTP");
    }
}

function validateUserId() {
    initRequest();
    req.onreadystatechange = processRequest;
    if (!target) target = document.getElementById("userid");
    var url = "validate?id=" + escape(target.value);
    req.open("GET", url, true);
    req.send(null);
}
```

URL is set to validate?id=greg



5. The ValidateServlet returns an XML document containing the results (Server)

```
public  void doGet(HttpServletRequest request, HttpServletResponse  response)
    throws IOException, ServletException {

 String targetId = request.getParameter("id");

 if ((targetId != null) && !accounts.containsKey(targetId.trim())) {
        response.setContentType("text/xml");
        response.setHeader("Cache-Control", "no-cache");
        response.getWriter().write("<valid>true</valid>");
    } else {
        response.setContentType("text/xml");
        response.setHeader("Cache-Control", "no-cache");
        response.getWriter().write("<valid>false</valid>");
    }
}
```

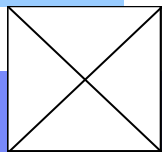


6. XMLHttpRequest object calls callback() function and processes the result

- **The XMLHttpRequest object was configured to call the processRequest() function when there is a state change to the readyState of the XMLHttpRequest object**

```
function processRequest() {
  if (req.readyState == 4) {
    if (req.status == 200) {
      var message = ...;
```

var message =
req.responseXML.getElementsByTagName("valid")[0].childNodes[0].nodeValue;

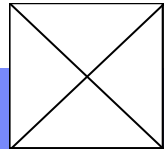


7. The HTML DOM is updated

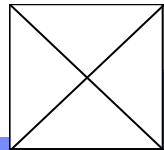
- JavaScript technology gets a reference to any element in a page using DOM API

- The recommended way to gain a reference to an element is to call
  - document.getElementById("userIdMessage"), where "userIdMessage" is the ID attribute of an element appearing in the HTML document

- JavaScript technology may now be used to modify the element's attributes; modify the element's style properties; or add, remove, or modify child elements



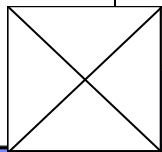
```
1. <script type="text/javascript">
2. function setMessageUsingDOM(message) {
3.     var userMessageElement = document.getElementById("userIdMessage");
4.     var messageText;
5.     if (message == "false") {
6.         userMessageElement.style.color = "red";
7.         messageText = "Invalid User Id";
8.     } else {
9.         userMessageElement.style.color = "green";
10.        messageText = "Valid User Id";
11.    }
12.    var messageBody = document.createTextNode(messageText);
13.    // if the messageBody element has been created simple replace it otherwise
14.    // append the new element
15.    if (userMessageElement.childNodes[0]) {
16.        userMessageElement.replaceChild(messageBody,
17.                                        userMessageElement.childNodes[0]);
18.    } else {
19.        userMessageElement.appendChild(messageBody);
20.    }
21.}
22.</script>
23.<body>
24.   <div id="userIdMessage"></div>
25.</body>
```



AJAX: DOM API & InnerHTML

- DOM APIs provide a means for JavaScript code to navigate/modify the content in a page

```
function setMessageUsingDOM(message) {
        var userMessageElement = document.getElementById("userIdMessage");
        var messageText;
        if (message == "false") {
            userMessageElement.style.color = "red";
            messageText = "Invalid User Id";
        } else {
            userMessageElement.style.color = "green";
            messageText = "Valid User Id";
        }
        var messageBody = document.createTextNode(messageText);
        if (userMessageElement.childNodes[0]) {
            userMessageElement.replaceChild(messageBody,
                userMessageElement.childNodes[0]);
        } else {
            userMessageElement.appendChild(messageBody);
        }
}
```

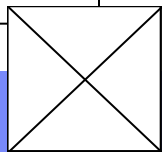


AJAX: DOM API & InnerHTML

- **Using innerHTML is easier: Sets or retrieves the HTML between the start and end tags of the object**

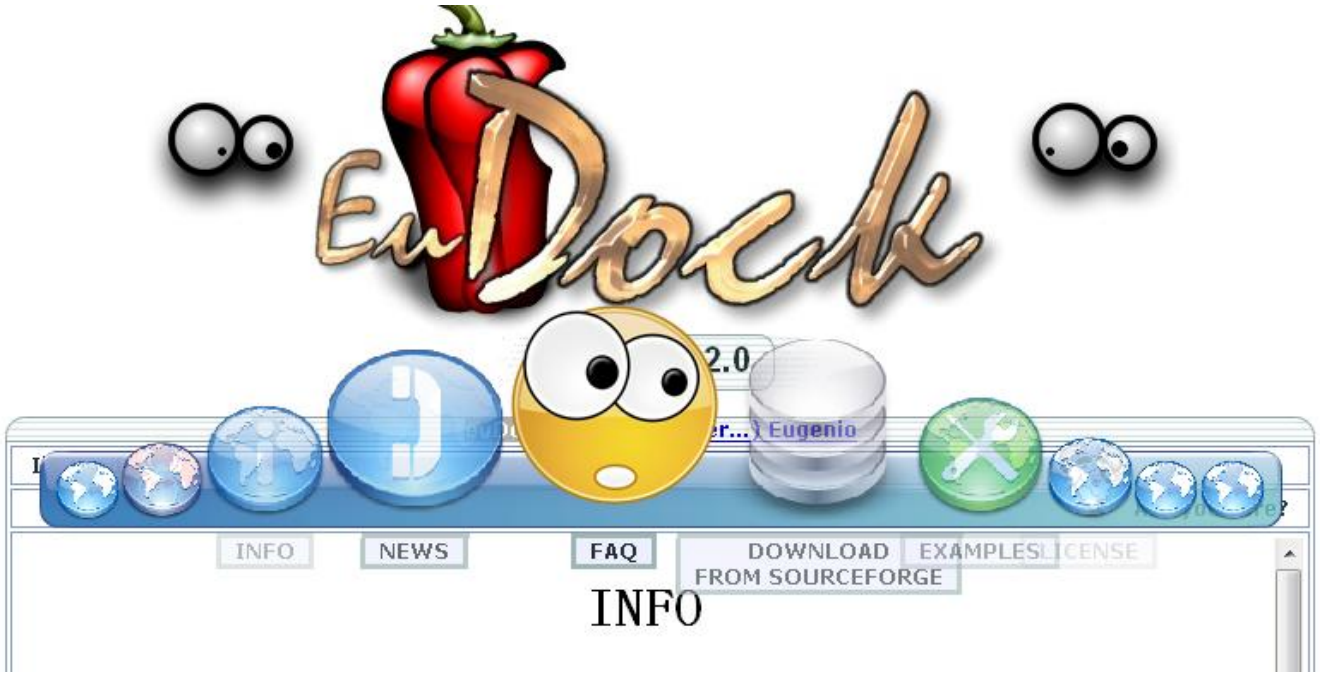
```
function setMessageUsingDOM(message) {
        var userMessageElement = document.getElementById("userIdMessage");
        var messageText;
        if (message == "false") {
            userMessageElement.style.color = "red";
            messageText = "Invalid User Id";
        } else {
            userMessageElement.style.color = "green";
            messageText = "Valid User Id";
        }
        userMessageElement.innerHTML = messageText;
}
```

Demo ajax-stock in tomcat



Ajax

- AJAX(Asynchronous JavaScript And XML)开发框架
  - GWT
  - DWR
  - DOJO
  - Openl
- Examp
  - Zimbr
    - http
  - dojo fi
    - http
  - euDoc..
    - http://eudock.jules.it/index-eudock2.0.php



Reference books

书名：Ajax基础教程
作者：(美)阿斯利森 舒塔
来源：人民邮电出版社
出版时间：2006年02月

书名：Ajax实战(Ajax in action中文版)
作者：克拉恩 帕斯卡雷洛 杰姆斯
来源：人民邮电出版社
出版时间：2006年04月

书名：Ajax Hacks(英文影印版)
作者：佩里
来源：东南大学出版社

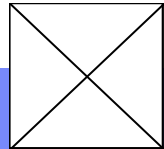
单页应用程序

- **Single Page Application（SPA）**

A **single-page application** (**SPA**), also known as **single-page interface** (**SPI**), is a web application or web site that fits on a single web page with the goal of providing a more fluid user experience akin to a desktop application.

需要考虑的问题

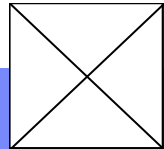
- 搜索引擎支持
- 浏览器的前进后退按钮支持
- 收藏(书签)
- 与服务器间大量的传输
- 多次通讯之间的逻辑问题



Mashup

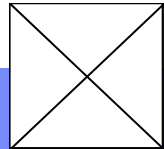
- 什么是 Mashup?
  - 最初的时候，Mashup 应用的含义只局限于在一个页面上集成了几个网站的内容
  - 发展成为对来自不同网站的信息进行处理，增加自己的业务逻辑，比如筛选过滤等等
  - Mashup 开发工具
    - IBM 的 QEDWiki
    - 微软的 Popfly
    - 雅虎的 Pipes

Mashup 是一种令人兴奋的交互式 Web 应用程序，它利用了从外部数据源检索到的内容来创建全新的创新服务。它们具有第二代 Web 应用程序的特点



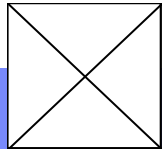
Mashup

- mashup 程序架构
  - API/内容提供者
    - 提供API
    - 屏幕抓取
      - 抓取者必须围绕一个源内容模型设计自己的工具，并且希望提供者一直采用这种模型来呈现内容
      - 缺少成熟的可重用屏幕抓取工具包软件
  - mashup 站点
  - 客户机的 Web 浏览器



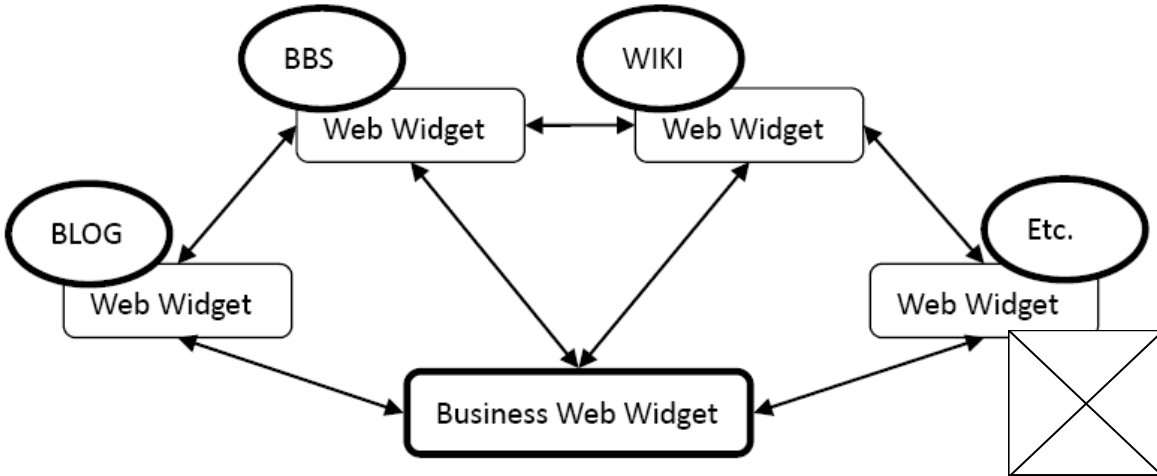
Mashup

- 技术挑战
  - 数据集成挑战：语义和数据的品质
  - 组件挑战
    - Ajax 引擎利用了一个 XMLHttpRequst 对象， Internet Explorer 6 中，这个对象是使用 ActiveX 实现的
  - 安全性问题
    - 调用时的身份认证
- 社会问题
  - 知识产权的保护



Mashup

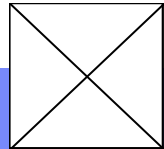
- **Mashup 类型**
  - **地图 mashup**
    - **Google Maps API Microsoft（Virtual Earth）、Yahoo（Yahoo Maps）和 AOL（MapQuest）**
  - **视频和图像 mashup**
    - **Flickr 使用自己的 API 来共享图像**
  - **搜索和购物 mashup**
    - **eBay 和 Amazon 之类的消费网站已经为通过编程访问自己的内容而发布了自己的 API**
  - **新闻 mashup**
    - **Diggdot.us**



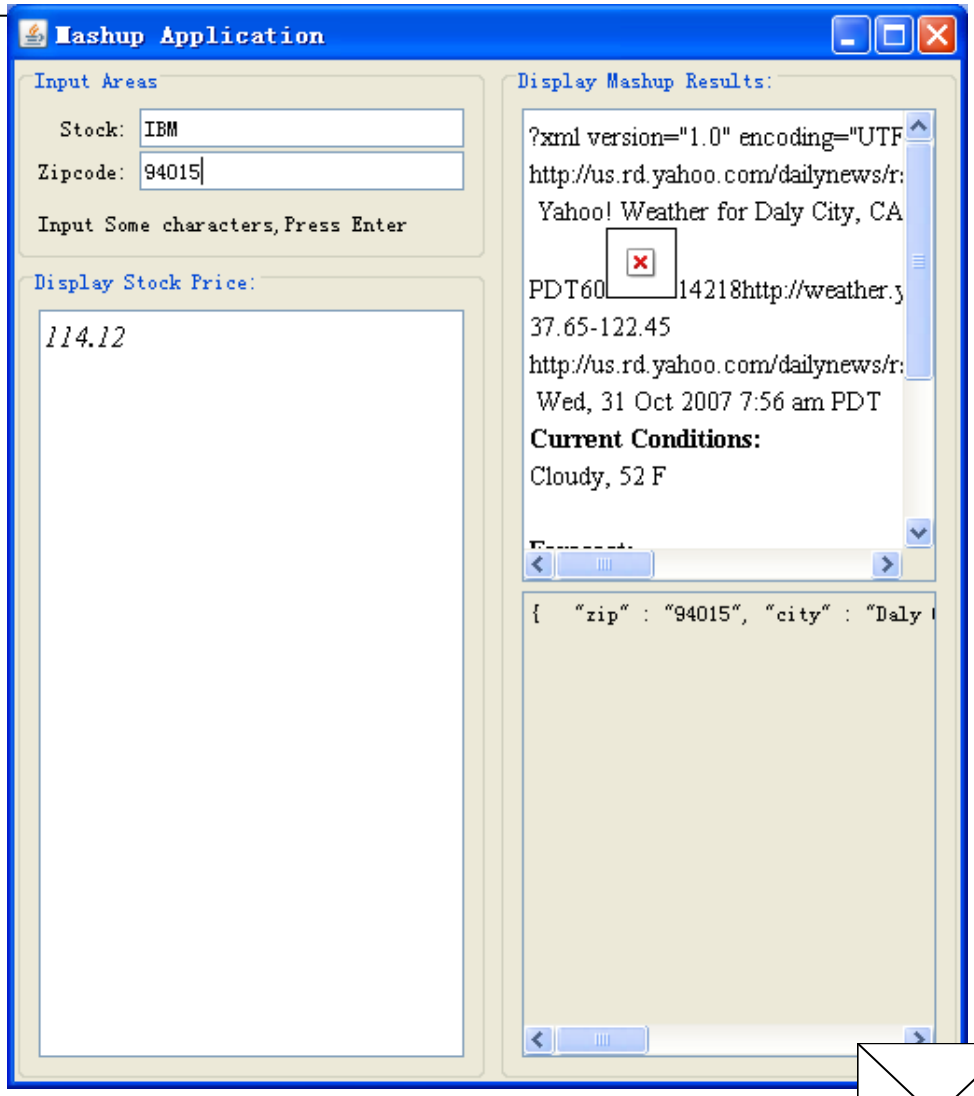
REST风格的Mashup

- **REST实例**
  - **IBM的股票价格**
    http://quote.yahoo.com/download/javasoft.beans?SYMBOLS=IBM&format=p“
  - **Yahoo! 的天气搜索的 API （zip code 94089）**
    http://weather.yahooapis.com/forecastrss?p=94089
  - **获得某个城市的信息**
    http://api.sunlightlabs.com/places.getCityStateFromZip.php?zip=94105&apikey=23kj234kjb3kjbcl87ds
    将返回一个 JSON 格式的结果数据

{ "zip" : "94105", "city" : "San Francisco", "state" : "CA" }



REST风格的Mashup



IBM_Demo_IBM_Mashup_Cente

Demo eclipse JavaFXHello项目

Web Game

- Travian

http://www.travian.cn/

- 三国风云
  http://sanguo.xiaonei.com/

Web Game

- **社交游戏**（Social game）:运行在SNS社区内