Local resource accessing mechanism on multiple mobile platform

Wei Shi[1],Minghui Wu[2*],Shuoping Wang[2],Min Guo[2],Bin Peng[2],Bin Ouyang[1],Tianzhou Chen[1]
[1]College of Computer Science and Technology, Zhejiang University
[2]City College, Zhejiang University
Hangzhou, Zhejiang, 310027, P.R.China
*corresponding author e-mail: mhwu@zucc.edu.cn

Abstract: **In recent years, with the advent of mobile Internet, Mobile Widget as an important element has been rapid development. It has become the mobile Internet hotspot. Mobile Widget to integrate Internet and local resources, available standard, HTML5, CSS (Cascading Style Sheets), and JavaScript Mobile Widget is the trend of technology development. But because of the different Mobile Widget platform used different standards, is not fully compatible with each other, so cross-platform development of mobile widget is still challenge, and one of the most difficult problems is implementation of cross-platform API. It makes Widget application for third party providers need to make a variety of different Widget for a business application to adapt to a different Widget platform. In order to solve this problem, this article proposes a cross-platform implementation scheme of local resource access Mobile Widget, experimental success and on different platforms.**

***Keywords:** Cross-Platform; Mobile Widget; JavaScript; HTML5*

I. INTRODUCTION

Application for Mobile Widget development in recent years was in full swing, with the rise of the mobile Web, application for Mobile Widget development has become main stream. Using standard HTML5 [1], CSS and JavaScript techniques to develop Mobile Widget is basically the current variety of intelligent mobile application development platform with unique technology. Mobile Widgets are developed using standard web technologies such as HTML, CSS, JavaScript and XML. These technologies are exactly those which are incorporated in the AJAX development approach [2]. In this sense, widgets could be regarded as a kind of AJAX application [3].

However, at present, international code for the Widget does not have a uniform; the result is that each product can only be running on their engines. For users, a Mobile Widget platform to run applications, you must first install the platform's Widget engine. To a certain extent, this is also creating problems for Mobile Widget Developer, Widget for a Widget engine developed products to be modified can only be run on a different Widget engines. Applications, especially Mobile Widget access to local resources is there is no uniform standard; fortunately the international W3C organization is in the process of developing a set of Widget code for series including work on Widget API interface standard. Can work out a solution of cross-platform Mobile Widget access to local resources? The answer is yes.

The rest of the paper is organized as follows. We introduce the related work in Section II.We described our work and compared with other people's work; we proposed our solution, the principle of our solution and Implementation on our solution in the Section III. In Section IV, we present results from the experiment. In the last section we draw a conclusion from advantage and disadvantage and described our future work.

II. RELATED WORK

There exist several commercial implementations of cross-platform mobile widget access local resource. Some companies like Nokia, Opera and JIL have been committed to developing cross-platform widget, but they are not very successful JIL can move ordinary widgets without APIs from one mobile platforms to another, and JIL Widgets runtime must be installed on target mobile platform; the same problems exists in other cross-platform development[4].

There exists work on mobile frameworks. For example, [5] introduced the most common cross-platform native frameworks. Such as Appcelerator Titanium, PhoneGap, Rhodes Rhomobile, Adobe AIR for Mobile Devices, Adobe Flex SDK “Hero” with Adobe Flash Builder “Burrito” and OpenPlug Elips Studio frameworks. The result of that thesis was also that the cross-platform native frameworks were at an early stage of their evolution in 2011.

Mobile Widget application to access local resources are mainly Native App and Web App and the Web App + Native App in three ways, Native App is above the platform level, ability to access and compatibility is better, can support online and offline message push or local resource access, camera and dial calls. But Native App of development cost to high many, maintained multiple version of update upgrade comparison trouble, user of installed threshold also comparison high, across platform comparison difficulties [6], Web App although support across platform and Terminal, and development cost low but message push sent enough timely and local resources called ability comparison weak; It is possible to take advantage of both web and native platforms by choosing to write hybrid applications. Typically web views can be included into native applications. Certain parts of the application, especially content view UI, can be written with web technologies. Smart phone platforms often provide APIs to make calls between native and JavaScript code. For Android the class to include a web view into a native Java application is android.webkit.WebView, whereas IOS uses UIWebView, and Windows Phone uses WebBrowser control [7].

The Browser-based mobile widget engine is compact and easy to develop using standard web technologies such as: HTML, CSS and JavaScript, so most of the program developers have the ability to develop such applications[8].Current mobile development platform to access local resources can be achieved mainly by China Mobile BAE (Browser based Application Engine) is a browser-based development platform. And completely free open source Openlaszlo [9] platform.

Although OpenLaszlo Supports cross-platform, it access to local resources is not strong; China Mobile BAE's shortcomings are: 1. China Mobile's currently BAE only support Android, Ophone, Symbian and Windows Mobile operate system; 2. China Mobile's Mobile Widget support JIL Widget standard is the standard, and the international W3C Widget standard or with varying degrees of difference;

Currently associated with the Mobile Widget of main standards are developed by the W3C, is W3C Widget 1.0 standard[10], joint innovation lab to develop the JIL Widget standards organization, the Open Mobile Terminal platform (OMTP) BONDI[11] standard, developed by the China Communications Standards Association (CCSA) developed Mobile Widget standard [12]. Luckily, The World Wide Web Consortium (W3C) addresses the problem of widget incompatibility and tries to standardize widget development within the Web Application Formats Working Group (WAF-WG). Relevant specification documents are: Widget Packaging and Configuration [13], Widget Packaging and Configuration (W3C Working Draft 21 February 2011) [14].

General coverage and authoritative W3C standards, and therefore conforms to W3C standards and cross-platform Mobile Widget to access local resources schemes are promising.

III. Our Work

A. Proposed Solution

In order to solve cross-platform Native App and Web App [15] problems to access local resources, the paper presents a solution of cross-platform Mobile Widget to access local resources. This scheme is based on standard Web technologies that is standard for HTML5, CSS and JavaScript technologies, through the browser as the core client-nested server side framework, a Mobile Widget between applications and devices to establish a communications hub, shielding platform differences for mobile devices. Consolidated using the JavaScript call local interfaces for mobile devices, so that JavaScript developers can call the mobile device address book, text and multimedia messages, camera, Bluetooth, GPS, WIFI, multimedia, database, and file system functions. So provide a cross-platform solution, truly a "written once, runs everywhere". In theory, Native App can access the interface can be invoked through the. It allow developer use either JavaScript, CSS and Web technologies such as HTML5 to achieve application Mobile Widget, but also through a unified interface to call a local interface. Combine the advantages of a Native App and Web App. The benefits of the scheme are:

1. Cross-platform, shield differences of mobile device platform, “written once, run everywhere”.

2. Direct access to local resources for mobile devices, JavaScript libraries, you can call the mobile device address book, text and multimedia messages, camera, Bluetooth, GPS, WIFI, multimedia, database, and file system functions.

3. Easy to use, this scheme fully with HTML5+CSS+JavaScript technology development, rich Internet applications can almost modify applications into Mobile Widget.

4. Can be applied to unify Mobile Widget provides effective and practical reference API standards, the scheme in full compliance with W3C standards, developers can use JavaScript Interface Pack, calling most of the local mobile device interface.

5. Scalability, extensibility of the scheme has a flexible; you can take advantage of existing mature JavaScript libraries, such as Ext JS, Dojo, jQuery Mobile, and more.

B. Principles of our solution

The first is to solve the problem of cross-platform. Mature uniform of the scheme using the standard Web technologies that is HTML5, CSS, and JavaScript technology. HTML5 not only inherited the advantages of HTML and make JavaScript to become a powerful language.HTML5 support Canvas draws directly, can get location information through the GeoLocation, To support local storage, and more. All of these make the Web application functionality and performance with Native applications on increasingly close. Through the HTML5 technology, can achieve real meaningful cross-platform application development. At the same time, no install, cross-platform features, will be the future trend of technology development. All major browser vendors support HTML5 standards gradually. HTML5 in the future is one of the standard cross-platform features. No doubt this scheme supports cross-platform creates the potential.

Secondly, access local resources and implementation in various Platform SDK and API without discrimination of interaction. The scheme aimed at different platforms extended the basic function of the browser components, making it into a powerful native API access mobile device browser. And then perform analysis through the browser, to implementation Presentation and business logic of the application UI Mobile Widget. Analysis in the process of implementation by calling the API JavaScript library, It completes and the indifference of the Platform SDK interaction and access to local resources. Different platforms used to implement call address book on your mobile device, text and multimedia messages, camera, Bluetooth, GPS, WIFI, multimedia, databases, and file systems, and other functions. The chart of the scheme implementation principle as below:

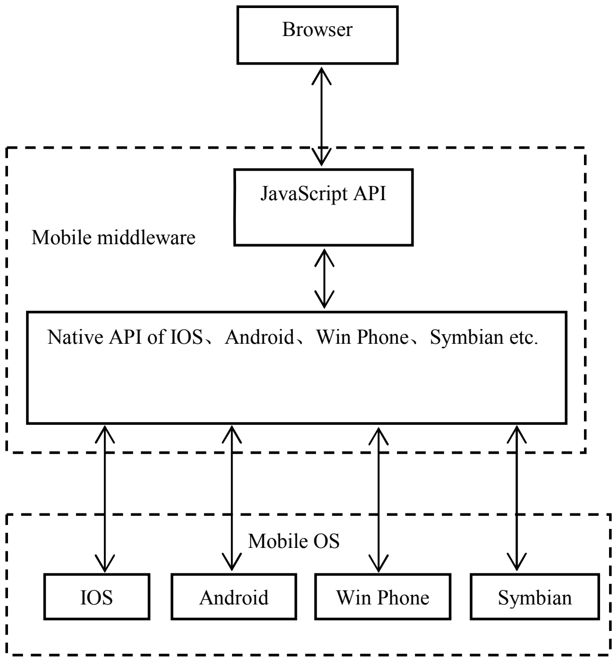


Figure1: the scheme principle

C. Implementation of our solution

It is well known that different mobile platform has built-in browser capabilities component. Basic functional components of the browser's ability to have a native API and mobile device bi-directional communication, we can be obtained by calling local JavaScript access to device API. We know the basic features of JavaScript and browser components of communication there are two ways, asynchronous communication (AJAX) and synchronous communication. AJAX called the "Asynchronous JavaScript and XML" (Asynchronous JavaScript and XML), is a kind of technology to create interactive web application and develop web. The biggest advantage to using Ajax is under the premise of maintaining data without updating the whole page, the term Ajax has come to represent a broad group of web technologies that can be used to implement a web application that communicates with a server in the background, without interfering with the current state of the page [16]. Every user action that normally would generate an HTTP request takes the form of a JavaScript call to the Ajax engine instead. Any response to a user action that doesn't require a trip back to the server—such as simple data validation, editing data in memory, and even some navigation—the engine handles on its own. If the engine needs something from the server in order to respond—if it's submitting data for processing, loading additional interface code, or retrieving new data—the engine makes those requests asynchronously, usually using XML, without stalling a user's interaction with the application[17].

JavaScript programming biggest problem comes from a variety of browser support for various technologies and standards. XmlHttpRequest Objects in different browsers have different ways of creating [18]. The scheme use a component has basic function of browser to render HTML, It uses a Plugin model to encapsulate the Native API, It covers basic functional components of the original browser method to implement a Web port for mobile devices in the form of local calls and mobile devices at the end of the local port to the Web port returns the results of asynchronous or synchronous calls. Encapsulate API on various mobile platforms. Here is the Encapsulate structure chart on the platform:

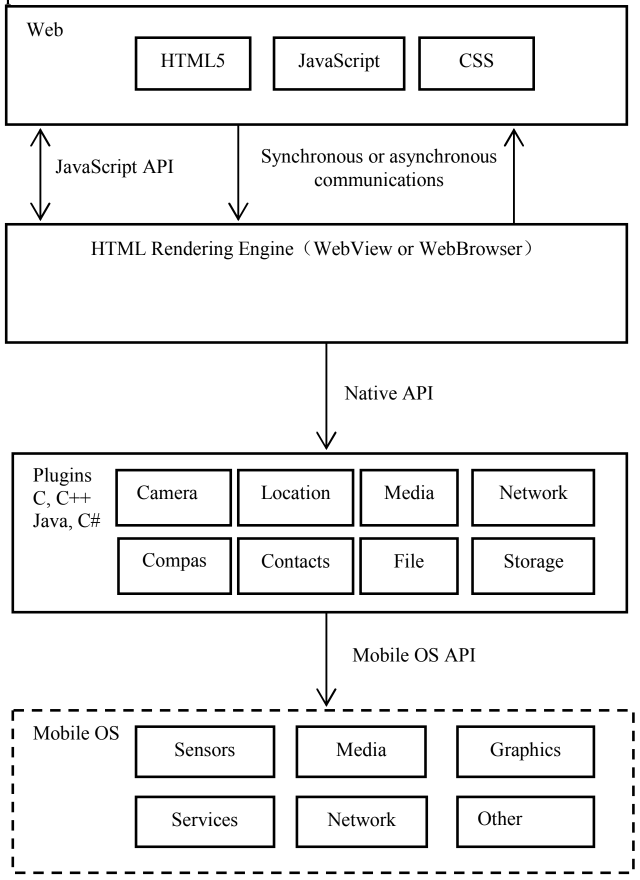


Figure2: the scheme implementation

Design JavaScript API for different mobile platforms, Includes the following API:

TABLE I. JAVASCRIPT API

API	Attribute	Methods	Arguments
Device	name platform version …	vibrate()	inMilliseconds
		cancelVibrate()	
		getInfo()	inInfoID
Geolocation	isLocationSupported	getCurrentPosition()	onSuccess, onError, onOption
		watchPosition()	onSuccess, onError,

		…	onOption
File	DirectoryEntry.name DirectoryEntry.fullpath File.name File.type File.size …	DirectoryEntry . moveTo() DirectoryEntry. copyTo() FileEntry.remove() …	parentEntry, newName, success, fail parentEntry, newName, success, fail success, fail …
…	…	…	…

Here is the JavaScript API Prototype:

Events: get Local events through JavaScript for mobile devices, such as Device is ready, press the menu key, press the back key, and so on.

Prototype of Function is: document.addEventListener ("deviceready", onDeviceReady, false);

Function onDeviceReady ()

{// after DeviceReady use the JavaScript API}.

Device: Get mobile device information through JavaScript, Such as the name of the device, platform, version, and so on.

Prototype of Function is:

var string = device.name;

var string = device.platform;

var string = device. Version;

Contacts: Get Local contact information through JavaScript for mobile devices, Such as local contact name, telephone, Email, and other properties, methods, such as add, find, and modify.

Prototype of Function is:

var string=contacts[i].Name;

var string=contacts[i].phonenumber;

var string=contacts[i].Email;

Contacts. Create ({"argue": "Value"});

Contacts.Find (contactfields, onSuccess, onError, options);

Contacts. Modify (contactfields, onSuccess, onError, options);

Function onSuccess (contacts)

{// handle the return result};

Function onError (Error)

{// handle the return result};

Because of space, omitted other API interface here.

IV. EXPERIMENTS

Due to limited space, we only describe the implementation on the representative Android and Win Phone Platform in this article.

A. implementation on Android

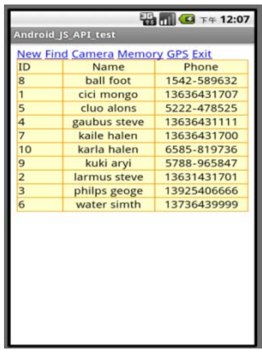
This scheme is based on WebView to render HTML for Android platform, It uses a Plugin model to encapsulate the Native API, It covers prompt method to implement a Web port to Android in the form of end of call, Using XHR or JSONP approach to achieve the Android port to the Web port returns the results of the asynchronous call.

Packaged by calling local JavaScript library called mobile device API, to achieve local resource access on mobile devices. Webview requires enabling JavaScript, the method is:

Webview.getSettings().setJavaScriptEnabled(true); A Java object can be bound to a JavaScript object use the addJavaScriptInterface method of Webview, So JavaScript objects can call Java methods, such as: Webview.addJavascriptInterface(new ContactsPlugin(), "contactsAction"); ContactsPlugin() is the Java class object, contactsAction is the JavaScript class object, After binding JavaScript objects and Java objects it will communicate. Development platform is the Eclipse; Mobile Widget portal program is a main page, such as index.html. It can be customized for other files of course. Also includes style files and JavaScript library files. Index.html file structure is generally as follows:

```
<html>
<head>
<Meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
<link rel="stylesheet" href="master.css" type="text/css" />
<title>Android_JS_API_test</title>
<script type="text/javascript"
src="jquery-1.7.1.js"></script>
<script "type="text/javascript"
src="AndroidAPI.js"></script>
</head>
<body onload="init()" >
<a href=" AddNewContact()">New</a>
<a href=" FindContact()">Find</a>
<a href=" Camera()">Camera</a>
<a href=" DisplayMemory()">Memory</a>
<a href=" GeoLocation()">GPS</a>
<a href=" Exit()">Exit</a>
<table border="1" width="100%" id="ContactTable"
cellspacing="0">
<tr>
<td width="20%"> ID</td>
<td width="40%" align="center">Name</td>
<td align="center">Phone</td>
</tr>
</table>
</body>
</html>
```

The actual results running on Android2.2 platform as follows:



ID	Name	Phone
8	ball foot	1542-589632
1	cici mongo	13636431707
5	cluo alons	5222-478525
4	gaubus steve	13636431111
7	kaile halen	13636431700
10	karla halen	6585-819736
9	kuki aryi	5788-965847
2	larmus steve	13631431701
3	philps geoge	13925406666
6	water simth	13736439999

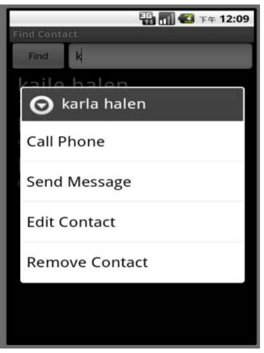


Figure3: the Contacts Figure4: Find Contact

B. implementation on Win Phone

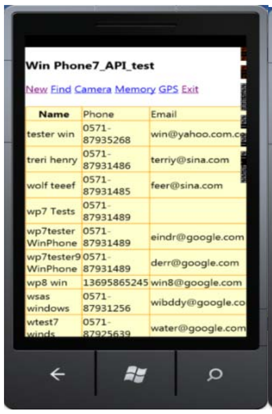
Win Phone7 Platform built-in browser component is WebBrowser,WebBrowser Components like Webview components and Android platforms, Both display a Web page can be resolved and can communicate with Javascript interaction in a Web page, So by calling local JavaScript access to device API.

Step one. JavaScript call Internal API of Win Phone7. Win Phone7 Provides the WebBrowser.ScriptNotify event, It allows Javascript call WebBrowser.ScriptNotify method when window.external.notify is called [19].WebBrowser.ScriptNotify Dynamic reflection of it through the parameters Command to respond to various needs.

Step two. Win Phone7 Provides webBrowser.InvokeScript method, Win Phone7 can call the method in JavaScript. Win Phone7 requires enable the JavaScript. The method is WebBrowser.IsScriptEnabled = true; the development platform is VS2010, Mobile Widget portal program is a main page, such as index.html, It can be customized for other names of course. Also includes style files, JavaScript Library files, and resource files, Index.html file structure is generally as follows:

```
<!doctype html>
<html>
<head>
<meta name="viewport" content="width=480;
user-scalable=no" />
<meta http-equiv="Content-type" content="text/html;
charset=utf-8"/>
<title>Win Phone7 Tests</title>
<link rel="stylesheet" href="master.css" type="text/css"
media="screen" charset="utf-8"/>
<script type="text/javascript" charset="utf-8"
src="WP7API.js"></script>
</head>
<body onload="init()" >
<h3>Win Phone7_API_test</h3>
<a href=" AddNewContact()">New</a>
<a href=" FindContact()">Find</a>
<a href=" Camera()">Camera</a>
<a href=" DisplayMemory()">Memory</a>
<a href=" GeoLocation()">GPS</a>
<a href=" Exit ()">Exit</a>
<table border="1" width="100%" id="Contactable"
cellspacing="0">
<tr>
<td width="30%"> Name </td><td width="30%"
align="center"> Phone </td>
<td align="left">Email</td>
</tr>
</table>
</body>
</html>
```

The actual results running on Win Phone7 platform as follows:

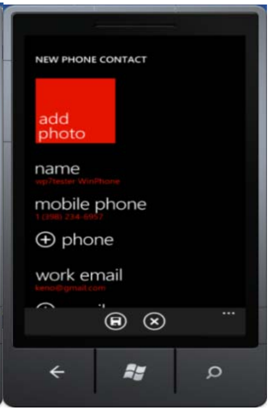


Win Phone7_API_test

New Find Camera Memory GPS Exit

Name	Phone	Email
tester win	0571-87935268	win@yahoo.com.c
treri henry	0571-87931486	terriy@sina.com
wolf teeef	0571-87931485	feer@sina.com
wp7 Tests	0571-87931489	
wp7tester WinPhone	0571-87931489	eindr@google.com
wp7tester9 WinPhone	0571-87931489	derr@google.com
wp8 win	13695865245	win8@google.com
wsas windows	0571-87931256	wibddy@google.co
wtest7 winds	0571-87925639	water@google.com

Figure5: the Contacts



NEW PHONE CONTACT

add photo

name
wp7tester WinPhone

mobile phone
1 (398) 234-6957

phone

work email
keno@gmail.com

Figure6: Add new Contact

V. CONCLUSION AND FUTURE WORK

In this paper, we present the concept how to achieve scheme of cross-platform mobile widget access local resource. This system is only proposed as a prototype, and has not completed all of the mobile widget functions that we can use.

By the above experiment, we can be sure of our proposal has a lot of advantages, 1. Cross platform 2. Direct access to local resources 3.Provide effective reference standard for unified Mobile Widget API. 4. Simple and useful. 5. Flexibility and scalability. But there are also some less than, such as, 1. Slow, mainly due to the WebKit browser kernel-dependent rendering resolution. 2. Lack of security for Mobile Widget, HTML5, JavaScript programs are written in plain text, prone to code tampering issues, so the security is not high. 3. Not convenient, because there is no integrated IDE development tool, it is not convenient. It does not support all platforms, there are some Special APIs: refers to functionality for just a particular platform, such as Logging API and Sensors API of WRT platform.

To this end, we intend to address deficiencies of the scheme optimization and improvement. First, we'd like to optimize browser rendering engine and take the technology of Application Cache. After the first download resources and code, you can cache the browser cache, so as to speed up the Application loading time code and resources. Second, for the Security issues for Mobile Widget, we intend to take the file encryption technology when write a Mobile Widget application. Mobile application with signing mechanism, classification control permissions and when come to danger it will remind the user timely to improve the security of the Mobile Widget. Finally, we'd like to explore the Mobile Widget integrated development environment based HTML5, with the Debug Mode.

ACKNOWLEDGEMENT

Our research work is partly supported by the Special Funds for Key Program of the China No. 2011ZX0302-004-002, the Science Foundation of Zhejiang Province under Grand No. 2010R50009, and Ministry of Education - INTEL Foundation of China under Grant No.MOE-INTEL-11-06.

REFERENCES

[1] http://www.mhtml5.com/

[2] J. J. Garret. Ajax: A New Approach to Web Applications. URL,http://www.adaptivepath.com/ideas/essays/archives/000385.php , February 2005.Accessed: September, 24th 2007.

[3] Christian Kaar, BSc An introduction to Widgets with particular emphasis on Mobile Widgets. Accessed: October 2007.

[4] Bin Zhang, et al: Research and Implementation of Cross-platform Development of Mobile Widget. Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on, on page(s): 146 -150

[5] Paananen, Timo: Smartphone Cross-platform Frameworks - A Case Study, JAMK University of Applied Science,Date:April 2011

[6] http://www.cnblogs.com/radom/archive/2012/03/05/2381155.html

[7] Sumi Helal, Raja Bose, and Wendong Li-Mobile Platforms and Development Environments,ISBN: 9781608458677,Synthesis Lectures on Mobile and Pervasive Computing #9 Print 1933-9011

[8] Yung-Wei Kao et al, A Cross-Platform Runtime Environment for Mo bile Widget-based Application, Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2011 International Conference on, Issue Date: 10-12 Oct. 2011,On page(s): 68 -71

[9] http://www.openlaszlo.org/node/416

[10] W3C. Widgets 1.0 Requirements. URL http://www.w3.org/tr/2007/wd-widgets-reqs-20070705/,July 2007. Accessed: September, 24th 2007.

[11] http://bondi.omtp.org/

[12] http://www.zte.com.cn/cndata/magazine/zte_communications/2011/2 /articles/201103/t20110323_225089.htm

[13] http://www.w3.org/2008/webapps/wiki/Main_Page#Widgets

[14] http://dev.w3.orgl2006/waf/widgets/(2011)

[15] Lynch, P.J, and Horton, S. Web Style Guide: Basic Design Principles for Creating Web Sites. New Haven and London: Yale University Press. 1999.

[16] http://en.wikipedia.org/wiki/Ajax_(programming)

[17] http://www.adaptivepath.com/ideas/ajax-new-approach-web-applicati ons

[18] http://en.wikipedia.org/wiki/ajax

[19] http://msdn.microsoft.com/en-us/library/system.windows.controls.we bbrowser.scriptnotify%28v=vs.95%29.aspx