The Validation of Credit Card Number on Wired and Wireless Internet

Chiyuan LI
Fire Engineering Dept., The Chinese People’s Armed Police Force Academy, Langfang, China
Email: chinafire_lcy@126.com

Zhiqiang YAO
Computer Science Dept., North China Institute of Aerospace Engineering, Langfang, China
Email: yaonciae@126.com

Abstract—Credit card number validation is a very common task performed by web application. This paper explores how LUHN Formula (Mod 10) can be applied on the wired and wireless internet to validate credit card number. JavaScript and VBScript are used separately with HTML to realize this algorithm on the wired internet and the internet explorer is used to present the test result; at the same time, WMLScript is used with WML to realize this algorithm on the wireless internet and WAP Toolkit simulator is used to present the test result. Lastly, conclusion is drawn about the widely application of this algorithm on the wired and wireless internet.

Index Terms—Credit Card Number Validation, LUHN Formula, JavaScript, VBScript, WMLScript, WAP Toolkit

I. INTRODUCTION

On-line shopping and e-bank[1][5] are more and more popular and credit card is widely used on the internet. During the operation of the credit card, input validation is a very common task performed by Web application. Imagine asking the user for his or her credit card number. If the user enters a series of digits randomly without fitting the standard of the credit card, the error could be detected early by client-side input validation, rather than posting the input to the server and cause a round-trip delay only to realize that the input is erroneous. At the same time, the wireless Internet is a new revolution upon us, one that will affect the world on a scale similar to that of the wired Internet. This is especially important for WAP(Wireless Application Protocol)[2] application, as the connection is inherently slow and therefore there is a necessity to reduce the number of connections to the server.

On the wired internet, besides using HTML(Hyper Text Markup Language)[2] for Web page rendering, Web developers also commonly use a scripting language like JavaScript or VBScript perform this client-side task; on the wireless internet[3], for WAP applications, the counterpart for WML(Wireless Markup Language) is the WMLScript language for client-side processing.

In this paper, scripting language JavaScript is used with HTML to realize the validation of credit card number on the wired internet; in addition, scripting language VBScript is used with HTML to realize the validation of credit card number on the wired internet; in order to realize the validation of credit card number on the wireless internet, scripting language WMLScript is used with WML.

II. LUHN FORMULA (MOD 10) ALGORITHM

Depending on the credit card type[4], most credit card numbers are encoded with a check digit. By running the credit card number through some algorithms, a check digit is often appended to the end of the credit card number. To validate that the number is a valid credit card number [6], the numbers are applied the same algorithm in the reverse manner.

The LUHN algorithm or LUHN formula[7][8], also known as the "modulus 10" or "mod 10" algorithm, is a simple checksum formula used to validate a variety of identification numbers, such as credit card numbers.

A. Formula (Mod 10) Algorithm Steps

1) Step 1: Multiply the value of alternate digits by 2, starting from the second rightmost digit;
2) Step 2: Add all the individual digits of the above products together with the un-doubled digits from the original number;
3) Step 3: If the total modulo 10 is equal to 0, then the number is valid according to the LUHN formula; otherwise it is invalid.

B. Illustration

Let us demonstrate this algorithm’s use with an example. In this example we will attempt to validate a credit card with the number 1234567890123456.

1) Step 1: First, multiply the value of the alternate digits by 2 starting from the second rightmost digit as described previously (See in Fig. 1).
2) Step 2:
Next, add the individual digits derived from the previous step. (Digits in parentheses are the products from Step 1):

(2) + 2 + (6) + 4 + (1 + 0) + 6 + (1 + 4) + 8 + (1 + 8) + 0 + (2) + 2 + (6) + 4 + (1 + 0) + 6 = 64

3) Step 3:
Finally, attempt to divide the resultant sum from step 2 by a value of 10.

Sum=64
Sixty-four is not divisible by 10; therefore the credit card number is invalid. The implementation of this algorithm using JavaScript, VBScript and WMLScript is shown in part III.

III. THE REALIZATION OF CREDIT CARD VALIDATION

The LUHN formula (Mod 10) algorithm is widely used on the wired internet and on the wireless internet. JavaScript and VBScript are used separately with HTML to implement this algorithm to realize the credit card number validation on the wired internet; at the same time, WMLScript is used with WML to implement this algorithm to realize the credit card number validation on the wireless internet.

A. The Application of LUHN Formula (Mod 10) Algorithm on The Wired Internet

1) The Application of LUHN Formula (Mod 10) Algorithm with JavaScript:[9][10]
One webpage file-----default_js.htm is designed for the customer to input the credit card number. When the user inputs the credit card number, and clicks the Confirm button, one function defined with the JavaScript language in the head of the file will work to validate the credit card number. As far as the source code is concerned, it is attached in the Appendix A.

The testing result is shown in Fig. 2 and Fig.3.

2) The Application of LUHN Formula (Mod 10) Algorithm with VBScript:
As the above webpage, VBScript[11][12] is used to validate the credit card number instead of JavaScript and the webpage file is default_vbs.htm. As far as the source code is concerned, it is attached in Appendix A.

The testing result is shown in Fig. 4 and Fig.5.

B. The Application of LUHN Formula (Mod 10) Algorithm on the Wireless Internet

The client software was developed using Nokia Wap Toolkit 2.0. This kit enables developers to create applications on the mobile phone based on the WAP (Wireless Application Protocol)[13]. Wireless Markup Language, (WML) is used to implement the Wireless Application Protocol (WAP) specification and creates pages that can be displayed in a WAP browser. WMLScript[13], a client-side scripting language, is used for developing tasks such as user input validation, generation of error message and other Dialog boxes etc. on the mobile phone. The Toolkit contains tools for editing both WML and WMLScript, and it can display the resulting code on a mobile service simulator (Fig.6).

As it is for developing Wireless Application Protocol(WAP) applications, the counterpart for Wireless Markup Language(WML) is the WMLScript language. The WMLScript language has similar objectives to those of JavaScript and VBScript.
WMLScript is a language that provides scripting capabilities to the WAP architecture. It complements the Wireless Markup Language. WMLScript is to WML what JavaScript is to HTML. In the context of WAP, WMLScript performs useful functions like user input validations and user prompts so as to reduce the round-trip delays to the origin server. WMLScript is loosely based on the ECMAScript (ECMA262) [15]. It is similar in syntax to scripting languages like JavaScript and JScript.

Firstly, one WML file----creditcard.wml, is designed for the user to input the credit card number from the mobile phone, the source code is attached in the Appendix B.

After user input the credit card number and press the confirm button, the function in the file of validate.wml will work. As we know, WMLScript [13] does not support the use of array, we have to improvise the use of array using a string. Since a string supports element indexing, we can manipulate the digits as though we are using an array. The source code for the validate.wmls is attached in the Appendix B.

2) The Testing of WML and WMLScript

Before the WMLScript is sent to the WAP device for execution, it has to be compiled (either by the WAP gateway[3] or through explicit compilation) into a binary format known as WMLScript bytecode. The WMLScript bytecode is then sent to the device to be interpreted by the WMLScript interpreter (located on the WAP device).

This compiled bytecode, together with the compiled WAP binary, will have to fit into the memory of the target device. Hence, the rule of minimizing the size of the WMLScript programs and WML decks [13] still holds.

For a WML deck to interact with WMLScript, it has to call functions in a WMLScript program defined with the extern keyword. The WMLScript program is stored in another file ending with the .wmls extension. Unlike Javascript, it cannot be embedded within the calling program----WML.

Now, we test these 2 files and Fig. 7 to Fig.13 show the resulting screens on the WAP Toolkit[15].
IV. CONCLUSION

It is both important and necessary to apply the credit card number validation in the web application. Although input validation task could be done on the server side, it is far more productive and efficient to perform the checking on the client side. Besides saving a round-trip to the server for the inputs to be validated, it allows your application more responsive to the user. No matter which scripting language is used, JavaScript or VBScript, LUHN formula (Mod 10) algorithm is realized well on the wired internet; at the same time, WML and WMLScript are used to test this algorithm on the wireless internet and testing result is satisfactory.

It is believed that credit card number validation algorithm will be widely used on the wired and wireless internet and the source codes provided in this article will be very instructive in the other related web applications [14] to enhance the client side functionality.

APPENDIX A: SOURCE CODES FOR CREDIT CARD VALIDATION ON THE WIRED INTERNET

File: default_js.htm

```html
<html>
<head>
<title>shopping on line</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<script language="JavaScript">
    function validate()
    {
        var number=document.form1.cardno.value.replace(/\D/g, '');
        // Set the string length and parity
        var number_length=number.length;
        var parity=number_length % 2;
        // Loop through each digit and do the maths
        var total=0;
        for (i=0; i < number_length; i++) {
            var digit=number.charAt(i);
            // Multiply alternate digits by two
            if (i % 2 == parity) {
                digit=digit * 2;
                // If the sum is two digits, add them together (in effect)
                if (digit > 9) {
                    digit=digit - 9;
                }
            }
        }
    }
</script>
</head>
<body>
<form name="form1">
    <input type="text" name="cardno" placeholder="Please Input Credit Card Number:"><br>
    <input type="submit" value="Confirm" onclick="validate()"><br>
    <input type="submit" value="Main" style="background-color: yellow; font-weight: bold; color: #000000; border: 1px solid black; padding: 5px; text-align: center;"
    value="Select"><br>
    <input type="submit" value="Back" style="background-color: yellow; font-weight: bold; color: #000000; border: 1px solid black; padding: 5px; text-align: center;">
</form>
</body>
</html>
```
} // Total up the digits total = total + parseInt(digit);

// If the total mod 10 equals 0, the number is valid if (total % 10 == 0) {
//document.form1.submit();
alert("Credit Card Number is valid");
return true;
} else {
alert("Credit Card Number is invalid");
return false;
}
</script>
</head>
<body>
<form name="form1" id="form1" method="post" action="getcard.asp">
Please input credit card No.: br/<input name="cardno" type="text" id="cardno" size="32">
<br />
<input type="button" name="Button" value="Confirm" onclick="validate()" />
</form>
</body>
</html>

File: default_vbs.htm
<html>
<head>
<title>shopping on line</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<script language="VBScript">
Sub validate()
num=document.form1.cardno.value
sum=0
Dim a
Redim a(21)
len1=Len(cstr(num))
For i=0 to len1-1
a(i)=mid(num,i+1,1)
Next
For i=0 to len1-1
If (len1 mod 2)<>0 then "credit card no. is odd
If (i mod 2)<>0 then
sum=sum+a(i)
else
temp=a(i)*2
sum=sum+mid(temp,1,1)
If len(temp)>1 then
sum=sum+mid(temp,2,1)
End If
End If
End If
Else
"credit card no is even
If (i mod 2)<>0 then
sum=sum+a(i)
else
temp=a(i)*2
sum=sum+mid(temp,1,1)
If len(temp)>1 then
sum=sum+mid(temp,2,1)
End If
End If
End If
Next
If(sum mod 10)>0 then
MsgBox "Credit Card No. is invalid"
Else
MsgBox "Credit Card No. is valid"
End If
End Sub
<script>
</head>
<body>
<form name="form1" id="form1" method="post" action="getcard.asp">
Please input credit card No.: <br />
<input name="cardno" type="text" id="cardno" size="32" value="1234567890123456" />
<br />
<input type="button" name="Button" value="Confirm" onclick="validate()" />
</form>
</body>
</html>

APPENDIX B: SOURCE CODES FOR CREDIT CARD VALIDATION ON THE WIRELESS INTERNET

File: creditcard.wml
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN" "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<template>
<do type="options" label="Main">
<go href="#card1" />
</do>
</template>
<card id="card1" title="Card 1">
<p>Please Input CreditCard Number:<input type="text" format="*N" name="ccno" />
<do type="accept" label="Confirm">
<go href="/validate.wmls#validate($ccno)" />
</do>
</p>
</card>
<card id="InvalidCard" title="Invalid">
<p>Credit Card Number is Invalid?
</p>
</card>
<card id="ValidCard" title="Valid">
<p>
</p>
</card>
</wml>

436 JOURNAL OF NETWORKS, VOL. 6, NO. 3, MARCH 2011

© 2011 ACADEMY PUBLISHER
Credit Card Number is Valid!
</p>
</card>
</wml>

File: validate.wmls
extern function validate(num)
{
var sum=0;
var temp,length;
length=String.length(num);
for(var i=length-1;i>=0;i--)
{
if(length%2!=0)
{
if(i%2!=0)
{
temp=Lang.parseInt(String.charAt(num,i))*2;
sum+=Lang.parseInt(String.charAt(temp,0));
if(String.length(temp)>1)
sum+=Lang.parseInt(String.charAt(temp,1));
}
else
sum+=Lang.parseInt(String.charAt(num,i));
}
else
{
if(i%2==0)
{
temp=Lang.parseInt(String.charAt(num,i))*2;
sum+=Lang.parseInt(String.charAt(temp,0));
if(String.length(temp)>1)
sum+=Lang.parseInt(String.charAt(temp,1));
}
else
sum+=Lang.parseInt(String.charAt(num,i));
}
if(sum%10!=0)
WMLBrowser.go("#InvalidCard"); //credit card no. is invalid.
else
WMLBrowser.go("#ValidCard"); //credit card no. is valid.
}

ACKNOWLEDGMENT

Zhiqiang YAO wishes to thank profoundly Mr.Zhiyuan AN, Dean of Computer Science Department, North China Institute of Aerospace Engineering for arranging his shift from the army college to the institution during winter semester of 2010 and for providing necessary support for teaching and research.

REFERENCES

Chiyuan Li, born in 1978, from Tianjin in China, Bachelor of Optical Engineering, Beijing Institute of Technology in 2001. Holder of a Combat Command master's degree from Institute of the Chinese People's Armed Police Force in 2010. Major field of study: fire command automation, data mining. He is a lecturer and works for Fire Engineering department of the Chinese People's Armed Police Force Academy from 2001 to present in Langfang City, Hebei Province. From 2007 to 2009, participation in China's "Eleventh Five-Year" national scientific and technological support projects "research on the regional fire dispatch field force organization and command technology"(Item Number:2006BAK06B05-2), designed and implemented a regional fire service emergency mobilization system.

Zhiqiang Yao, born in 1971, earned master degree in Tsinghua University in 2000, an MCSD(Microsoft Certified Solution Developer).
From 2003 to 2009, he worked at Scientific Research Institute of the Chinese People's Armed Police Force Academy. Since then, he worked at Computer Science Department of North China Institute of Aerospace Engineering and taught computer basics, database and other courses.