Automated Server-side Form Validation

Abstract— **The main goal of this article is to introduce the new technique of server-side HTML form validation. For every custom-built web application, the developer has to write their own validation code for every form used in application. This is a time consuming work for the developers which kills a lot of the development time of the application. This technique will help the developers to validate their forms without writing validation code which will minimize the development time for web applications and as well as increase developer's output. To validate a HTML form's field server-side validation is important because client-side validation is done by JavaScript. JavaScript validation will be meaningless if the users disable the JavaScript function of their browser. That's why server-side validation must be done by the developer in each web application. Its application procedure in web based applications is also discussed.**

Keyword— **application, automated, form, server-side, technique, validation, web**

I. Introduction

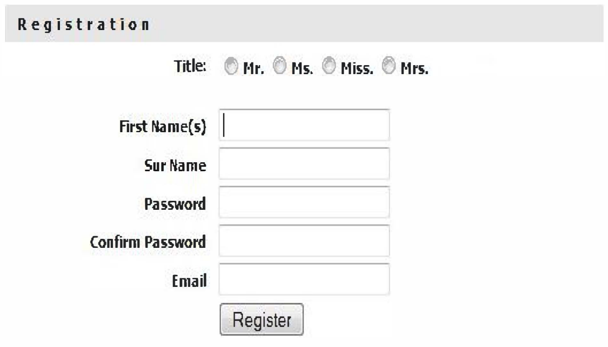
A web application is an application that is accessed over a network such as the Internet or an intranet [1]. Here an intranet can be LAN, MAN, CAN, etc. This application is hosted on local server or in web server. Different browsers are used to access it. Web applications are very much popular among Internet users. These applications are developed by using different web languages like PHP, ASP, JSP, etc. Most of application take user's input through HTML form and stored it in their database. Many of these text fields and forms used in web apps are straightforward; some are more challenging for the user to correctly interpret and fill [2]. Both technical and non-technical users access those forms. They may not able to give right value in each field of the forms. An essential part of form handling at the developer's end is the checking and validating of the input, before it can be added to a database or used as basis for other calculations [2]. For some special apps, developers have to write complex business logic functions to validate data given in a field of a form. For every single application, they have to write this code again and again. This technique can be implemented by developing an application program interface (API) which is installed by the coder at the server end to validate each form of the web apps at server-side. Here coder has to introduce each HTML form of the apps to the API. The API will able to read each field of HTML form and store the validation information in a database. Then this information will be used for validation.

II. HTML Form Validation

Form validation is the process of checking the data that a user enters into form fields [3]. There are mainly two types of validation related to web applications done by developers. One is client-side and another is server-side validation. Client-side validation uses a scripting language, like JavaScript or VBScript, to check the data that a user enters into a form before the web browser sends that data to the server for processing. Client-side validation is usually very quick in responsiveness. However, it may not be entirely reliable, given that users may choose to disable JavaScript. On the other hand, server-side validation is somewhat slower that client-side validation and relies on a program located on the server. While server-side validation may be slower, it is very reliable. The user can't disable the server-side validation script [3]. So developer must use server-side validation to validate the data given in the form by the users.

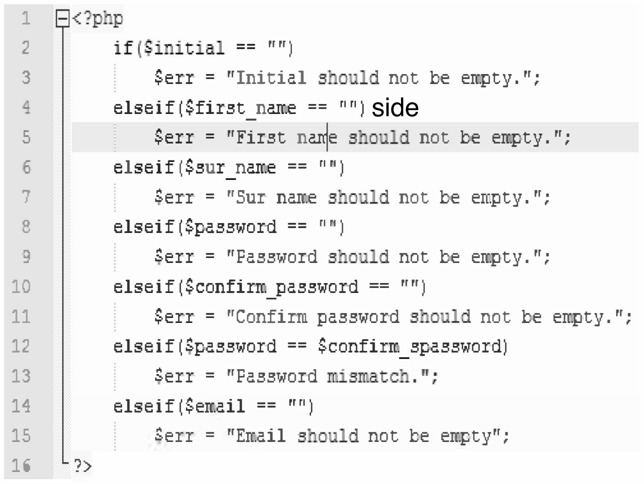
III. Current server-side validation trends

At present every developers has to write their own server-side validation code or use of validation class to validate each form in the developed applications. They have to write some repeated code for some fields in different forms or use function calls. For example, consider the registration form as shown in Fig.1 and its validation code using PHP which is one of the web-based programming languages is shown in Fig.2.



Registration	
Title:	○ Mr. ○ Ms. ○ Miss. ○ Mrs.
First Name(s)	
Sur Name	
Password	
Confirm Password	
Email	
	Register

Fig.1. Screenshot of Registration Form in HTML



```
<?php
    if($initial == "")
        $err = "Initial should not be empty.";
    elseif($first_name == "") side
        $err = "First name should not be empty.";
    elseif($sur_name == "")
        $err = "Sur name should not be empty.";
    elseif($password == "")
        $err = "Password should not be empty.";
    elseif($confirm_password == "")
        $err = "Confirm password should not be empty.";
    elseif($password == $confirm_spassword)
        $err = "Password mismatch.";
    elseif($email == "")
        $err = "Email should not be empty";
?>
```

Fig.2. Validation code of the registration form in PHP

There are six fields in the registration form shown in Fig.1. Here each field is mandatory. So we have to write validation code for each field as shown in Fig.2. Here validation is done is based only the field is empty or not where line of code (LOC) is only 14. But it's not enough

978-1-4673-1154-0/12/$31.00 ©2012 IEEE ICIEV 2012

for server-side validation. The basic validation includes [2]:

- **Required field validation:** Checking that required fields contain a value.
- **Type check:** Checks whether numeric field only holds numeric or date fields contains only dates or string field contains strings.
- **Range check:** Elements length can be checked.
- **Format check:** Text elements may define regular expressions the input must uphold.

So to check all of the above criteria for our registration form, we have to write the code where LOC will be about 56.

IV. AUTOMATED SERVER-SIDE FORM VALIDATION TECHNIQUE

This validation is called 'automated' validation technique because by applying this technique, coder will be able to validate their forms in the web apps without writing server-side validation code. The primary concern of this technique is to able parse the web pages containing the form and find out all the elements exist in the form for validation. This structural information of a webpage can be obtained from the DOM (Document Object Model) tree [4], which is maintained by each web browser when it loads any web page. DOM is a model of programming that concerns the way in which we represent objects contained in a single web page. There are different levels of the DOM Standard, as proposed by the W3C [3]. Here we use only level-1 called DOM1 standard. Fig.3 shows various processing modules of this technique. The strategies followed in this technique are described in the following steps as shown in Fig.3.

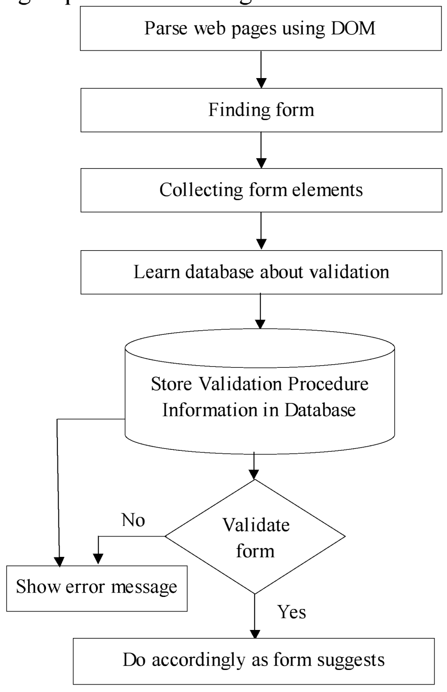

Parse web pages using DOM
Finding form
Collecting form elements
Learn database about validation
Store Validation Procedure Information in Database
Validate form
No
Yes
Show error message
Do accordingly as form suggests

Fig.3. Processing block of this technique.

A. Parse Web Pages Using DOM

The main challenge is here to parse the web pages containing form in the web application using simple HTML DOM parser. First we will create a DOM object from the file or the URL of the submitted form. For every DOM node, our script works with the tagname properties associated the DOM element and parse every required element of the form.

B. Finding Form

Next our attention is to find out the <form> tag from this page by skipping all other tag before it and continue searching until find </form> tag. The contents exist between these two tags is our form elements which are needed to validate. Here developer will give each form a name to uniquely identify them in the web apps.

C. Collecting Form Elements

A form in a webpage may contain different types of tags, labels and legends. Here, DOM will collect only the element containing the tag name <input>, <select> and <textarea>. Because these tags contain the data which needed to validate before further processing. Here input tag having 'submit' and 'button' type will be skipped during collection.

D. Learn Database about Validation

After collecting every form elements, consider only those elements which are needed to validate. For example consider the HTML form of Fig.4 for a personal profile.

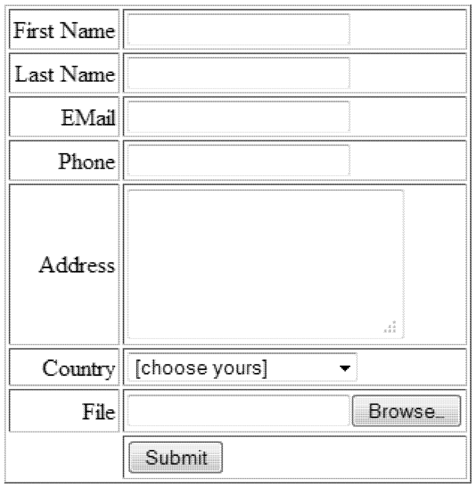
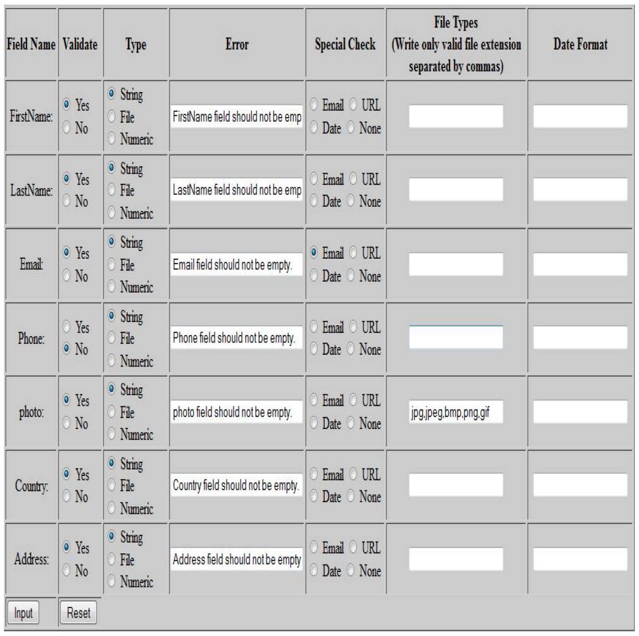


Fig.4. Processing block of this technique.

In this technique, when the developer runs the page containing form for the first time and submit it, he/she will get the interface shown in Fig.5 after processing this form using DOM. This interface is used to specify which field of the form is to validate and which will be the logic behind the validation. Here the coder has to set up the following options for validation:

- Validate the field or not.
- Types of value accepted by the field. (e.g numeric, text, file, etc)
- Special check like email, date and URL validation is required for that field or not.
- Error message to be displayed for corresponding field.

- If the field is file types then specify the file extension for accepted files types.
- If the field is date types then specify the required date format like (dd-mm-yy, dd-mm-yyyy, etc).



Field Name	Validate	Type	Error	Special Check	File Types (Write only valid file extension separated by commas)	Date Format
FirstName:	[✓ Yes] No	[✓ String] File Numeric	FirstName field should not be emp	Email URL Date None		
LastName:	[✓ Yes] No	[✓ String] File Numeric	LastName field should not be emp	Email URL Date None		
Email:	[✓ Yes] No	[✓ String] File Numeric	Email field should not be empty.	[✓ Email] URL Date None		
Phone:	Yes [✓ No]	[✓ String] File Numeric	Phone field should not be empty.	Email URL Date None		
photo:	[✓ Yes] No	[✓ String] File Numeric	photo field should not be empty.	Email URL Date None	jpg,jpeg,bmp,png,gif	
Country:	[✓ Yes] No	[✓ String] File Numeric	Country field should not be empty.	Email URL Date None		
Address:	[✓ Yes] No	[✓ String] File Numeric	Address field should not be empty	Email URL Date None		
Input	Reset					

Fig.5. Interface after processing above form using DOM.

E. *Store Validation Information After Learned*

The validation information related to each field of the form will then be stored in the database of the corresponding web applications where this technique will be applied. The E-R diagram of the required database for this learning procedure is show in fig.6.

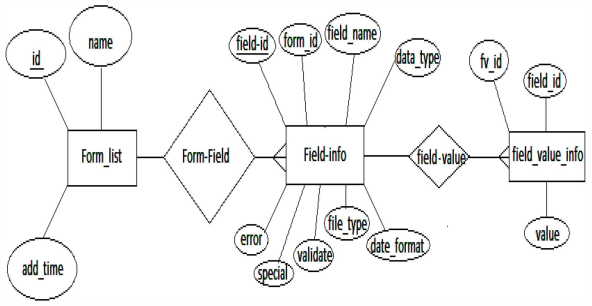


Fig.6. E-R diagram of the required database for validation.

Here, the every form data is stored in the form_list table. Each field's information of every form goes to field-info table. Some special field like radio button, checkbox, selectbox have multiple values. These values are stored in the field_value_info table.

F. *Validation*

Here, validation is occurred in the server-side. When the user runs the pages containing the form again, the form is validated using the validation criteria stored in the database for that form.

V. IMPLEMENTATIONS

This technique can be applied to any web applications developed by using one server side language like PHP, ASP, JSP, etc and MYSQL, SQL server, etc as a database. Here we can choose PHP and MYSQL. When this technique is implemented, we called it API. This API will work in two phases. One is learning phase and another is validation phase.

A. *Learning Phase*

In this phase, the API is installed in the web application by the developer. After installed, developer run every page containing form of the web apps and submit it without giving any data. Our API will automatically track the submission and generate a form using the elements like (input, select, textarea) used in that form. Then coder will then select from the form which fields are needed to validate or not as discussed above [section III (D)]. Here API code interacts with the existing web application by a special configuration tool of apache web server called '.htaccess' [5].

B. *Validation Phase*

In this phase, once API learned about a form's field for validation then in the next run of the form submission, server side form validation will work automatically. If any field is invalid, then error message given by the coder during learning phase will be displayed.

VI. **RELATED WORK**

Research in form processing has been done by few researchers for both on the desktop and in web-based applications. There is no researcher who has worked on automated server-side form validation in PHP language. Most of the researchers worked for client side validation in the area of web applications. PowerForms [4] support client-side validation for user. XForms [5] is an XML application that represents the next generation of forms for the Web. XForms accommodates form component reuse, fosters strong data type validation, and thus reduces the need for scripting [5]. Validator Toolkit [6] provides a set of validators for the ASP.NET MVC framework to validate HTML forms on the client and server-side using validation sets. HTML5 [7] has several new input types for forms. These new features allow better input control and validation. But this validation is all about client-side validation which is not that much strong as server-side validation. For example – in HTML5 to validate an email field we have to take new email type for input tag. This field shows error if the email value is like abc.com, abc, etc. But it accepts value like abc@cde which is not at all a valid email address.

VII. **LIMITATION**

This technique may not work for some special validation of a text field like telephone number because various country follow different format of their telephone number. Besides this, it will not work for those server-side validations where business logic is needed to implement to get a particular field value.

VIII. **CONCLUSION**

This technique is a different approach to server-side form validation from our early researchers which work automatically with the web applications. Perfect validation is very important for efficient form data processing. This technique will really minimize the development time of web-based software validation for server-side.

ACKNOWLEDGEMENTS

I would like to thank to everyone who has given me the opportunity to work as a web developer and then in teaching. This has helped me a lot to think about web-based application validation problems.

REFERENCES

[1] Web App - Wikipedia, the free encyclopedia, Available from: http://en.wikipedia.org/wiki/Web-app

[2] M. Bohoj, N.O. Bouvin, and H. Gammelmark, "*AdapForms: A Framework for Creating and Validating Adaptive Forms*", in Proc. ICWE, 2011, pp.105-120.

[3] CGI Extremes Tutorials - .Htacess .Htaccess and Password Protection, Available: http://www.cgiextremes.com/extras/Tips_Tutorials/htaccess.html

[4] C. Brabrand, Moller, A. M. Ricky, and Schwartzbach, "*Powerforms: Declarative client-side form field validation*", World Wide Web 3(4), 205-214 (2000).

[5] W3C: XFORMS 1.1 (2007), Available: http://www.w3.org/TR/xforms11/

[6] J. Baurle, "*Validator Toolkit for ASP.NET MVC*", Available: http://mvcvalidatortoolkit.codeplex.com/

[7] HTML5 New Input Types, Available from: http://www.w3schools.com/html5/html5_form_input_types.asp