Regular Expressions

Chapter 12

Randy Connolly and Ricardo Hoar

Fundamentals of Web Development

Section 4 of 6

REGULAR EXPRESSIONS

Regular Expressions

May seem irregular at first glance

A **regular expression** is a set of special characters that define a pattern.

They are a type of language that is intended for the matching and manipulation of text.

Regular expressions are a concise way to eliminate the conditional logic that would be necessary to ensure that **input data follows a specific format**.

Regular Expressions

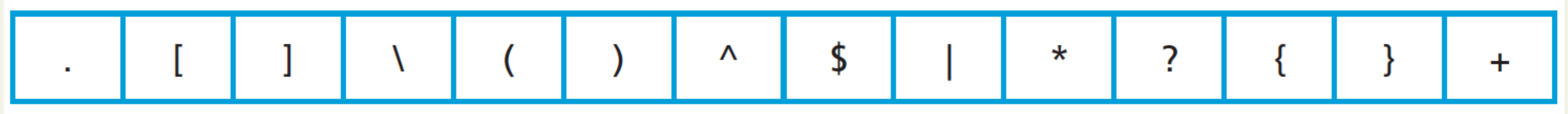
Syntax

A regular expression consists of two types of characters: **literals** and **metacharacters**.

- A **literal** is a character you wish to match in the target
- A **metacharacter** is a special symbol that acts as a command to the regular expression parser

Regular Expressions

Characters with Special Meaning



.	[	]	\	(	)	^	$	|	*	?	{	}	+

TABLE 12.2 Regular Expression Metacharacters (i.e., Characters with Special Meaning)

To use a metacharacter as a literal, you will need to escape it by prefacing it with a backslash (\)

Regular Expressions

Expression	Description
^ … $	If used at the very start and end of the regular expression, it means that the entire string (and not just a substring) must match the rest of the regular expression contained between the ^ and the $ symbols.
\t	Matches a tab character.
\n	Matches a new line character.
.	Matches any character other than \n.
[qwerty]	Matches any single character of the set contained within the brackets.
[^qwerty]	Matches any single character not contained within the brackets.
[a-z]	Matches any single character within range of characters.
\w	Matches any word character. Equivalent to [a-zA-Z0-9].
\W	Matches any nonword character.
\s	Matches any white-space character.
\S	Matches any nonwhite-space character.
\d	Matches any digit.
\D	Matches any nondigit.
*	Indicates zero or more matches.
+	Indicates one or more matches.
?	Indicates zero or one match.
{n}	Indicates exactly n matches.
{n,}	Indicates n or more matches.
{n,m}	Indicates at least n but no more than m matches.
|	Matches any one of the terms separated by the | character. Equivalent to Boolean OR.
()	Groups a subexpression. Grouping can make a regular expression easier to understand.

Regular Expressions

Building one example

Consider a regular expression that would match a North American phone number without the area code.

A valid number contains **three numbers, followed by a dash, followed by four numbers without any other character.**

The regular expression for this would be:

```
^\d{3}–\d{4}$
```

Regular Expressions

three numbers, followed by a dash, followed by four numbers

```
^\d{3}–\d{4}$
```

- The **dash** is a literal character; the rest are all metacharacters
- The **^** and **$** symbol indicate the beginning and end of the string, respectively
- The metacharacter **\d** indicates a digit, while the metacharacters **{3}** and **{4}** indicate three and four repetitions of the previous match (i.e., a digit), respectively

Regular Expressions

three numbers, followed by a dash, followed by four numbers

A more sophisticated regular expression for a phone number would **not allow the first digit in the phone number to be a zero ("0") or a one ("1").**

The modified regular expression for this would be:

```
^[2-9]\d{2}–\d{4}$
```

Regular Expressions

Any number (but 0,1), then 2 more, a dash and 4 more.

^[2-9]\d{2}–\d{4}$

- The **[2-9]** metacharacter indicates that the first character must be a digit within the range 2 through 9
- Since only two more numbers are needed the pattern **\d{3}** becomes **\d{2}**

Regular Expressions

Allow a space, period, or dash in the number.

We can make our regular expression a bit more flexible by allowing either a single space (440 6061), a period (440.6061), or a dash (440-6061) between the two sets of numbers.

We can do this via the [] metacharacter:

```
^[2-9]\d{2}[–\s\.]\d{4}$
```

Regular Expressions

Allow a space, period, or dash in the number.

```
^[2-9]\d{2}[–\s\.]\d{4}$
```

This expression indicates that the fourth character in the input must match one of the three characters contained within the square brackets (**–** matches a dash, **\s** matches a white space, and **\.** matches a period)

Regular Expressions

Allow multiple spaces

If we want to **allow multiple spaces (but only a single dash or period)** in our number:

^[2-9]\d{2}[–\s\.]\s*\d{4}$

The metacharacter sequence **\s*** matches zero or more white spaces.

Regular Expressions

How about area code

To allow the area code to be

• Surrounded by Brackets	(403) 440-6061
• Separated with spaces	403 440 6061
• A Dash	403-440-6061
• A Period	403.440.6061

```
^\(?\s*\d{3}\s*[\)–\.]?\s*[2-9]\d{2}\s*[–\.]\s*\d{4}$
```

Regular Expressions

How about area code

```
^\(?\s*\d{3}\s*[\)–\.]?\s*[2-9]\d{2}\s*[–\.]\s*\d{4}$
```

The expression now matches

- zero or one "(" characters **\(?**
- zero or more spaces **\s*** three digits **\d{3}**
- zero or more spaces **\s***
- either a ")" a "-", or a "." character **[\)-\.]?**
- zero or more spaces **\s***

(403) 440-6061

403 440 6061

403-440-6061

403.440.6061

Regular Expressions Alternative

`^(\(?\s*\d{3}\s*[\)–\.]?\s*)?[2-9]\d{2}\s*[–\.]\s*\d{4}$`

```
var phone=document.getElementById("phone").value;
var parts = phone.split(".");                // split on .
if (parts.length !=3){
   parts = phone.split("-");                 // split on -
}
if (parts.length == 3) {
   var valid=true;                  // use a flag to track validity
   for (var i=0; i < parts.length; i++) {
      // check that each component is a number
      if (!isNumeric(parts[i])) {
         alert( "you have a non-numeric component");
         valid=false;
      } else { // depending on which component make sure it's in range
         if (i<2) {
            if (parts[i]<100 || parts[i]>999) {
               valid=false;
            }
         }
         else {
            if (parts[i]<1000 || parts[i]>9999) {
               valid=false;
            }
         }
      }
      } // end if isNumeric
      } // end for loop
if (valid) {
   alert(phone + "is a valid phone number");
}
}
alert ("not a valid phone number");
```

LISTING 12.7 A phone number validation script without regular expressions

Some Common Regular Expressions

Regular Expression	Description
^\S{0,8}$	Matches 0 to 8 nonspace characters.
^[a-zA-Z]\w{8,16}$	Simple password expression. The password must be at least 8 characters but no more than 16 characters long.
^\d{5}(-\d{4})?$	American zip code.
^((0[1-9])|(1[0-2]))\/(\d{4})$	Month and years in format mm/yyyy.
^(.+)@([^\.].*)\.([a-z]{2,})$	Email validation based on current standard naming rules.
^((http|https)://)?([\w-] +\.)+[\w]+(/[\w- ./?]*)?$	URL validation. After either http:// or https://, it matches word characters or hyphens, followed by a period followed by either a forward slash, word characters, or a period.
^4\d{3}[\s\-]d{4}[\s\-] d{4}[\s\-]d{4}$	Visa credit card number
^5[1-5]\d{2}[\s\-]d{4}[\s\-] d{4}[\s\-]d{4}$	Mastercard credit card number

Regex is everywhere

Including MySQL

MySQL also supports regular expressions through the REGEXP operator.

For instance, the following SQL statement matches all art works whose title contains one or more numeric digits:

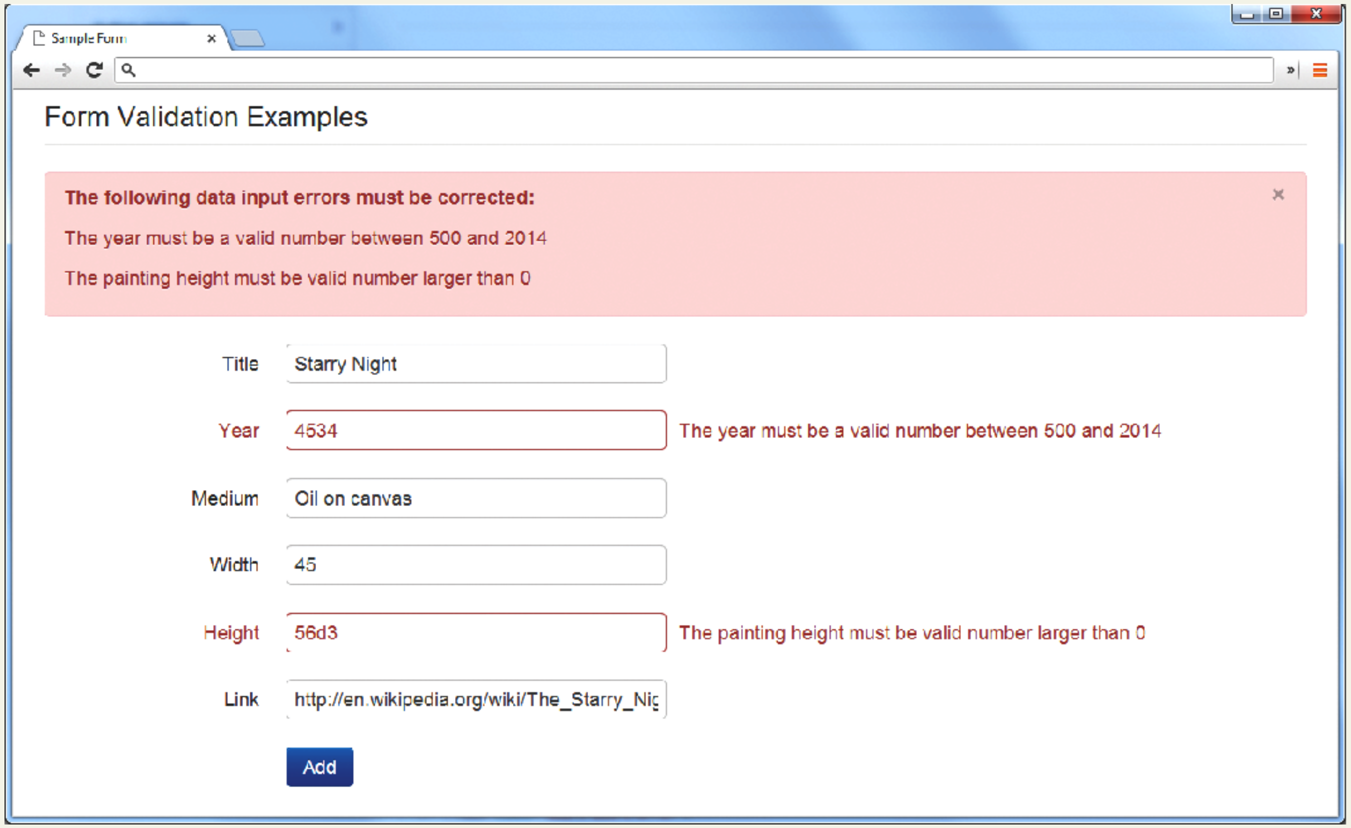
```
SELECT * FROM ArtWorks WHERE Title REGEXP '[0-9]+'
```

Section 5 of 6

VALIDATING USER INPUT

Notifying the User

What's wrong, where is it, and how to fix it.



Form Validation Examples

The following data input errors must be corrected:

The year must be a valid number between 500 and 2014

The painting height must be valid number larger than 0

Title	Starry Night	
Year	4534	The year must be a valid number between 500 and 2014
Medium	Oil on canvas	
Width	45	
Height	56d3	The painting height must be valid number larger than 0
Link	http://en.wikipedia.org/wiki/The_Starry_Nig	

Types of Input Validation

- **Required information**. Some data fields just cannot be left empty. For instance, the principal name of things or people is usually a required field. Other fields such as emails, phones, or passwords are typically required values.
- **Correct data type**. Some input fields must follow the rules for its data type in order to be considered valid.
- **Correct format**. Some information, such as postal codes, credit card numbers, and social security numbers have to follow certain pattern rules.

Types of Input Validation

Continued

- **Comparison**. Perhaps the most common example of this type of validation is entering passwords: most sites require the user to enter the password twice to ensure the two entered values are identical.
- **Range check**. Information such as numbers and dates have infinite possible values. However, most systems need numbers and dates to fall within realistic ranges.
- **Custom**. Some validations are more complex and are unique to a particular application

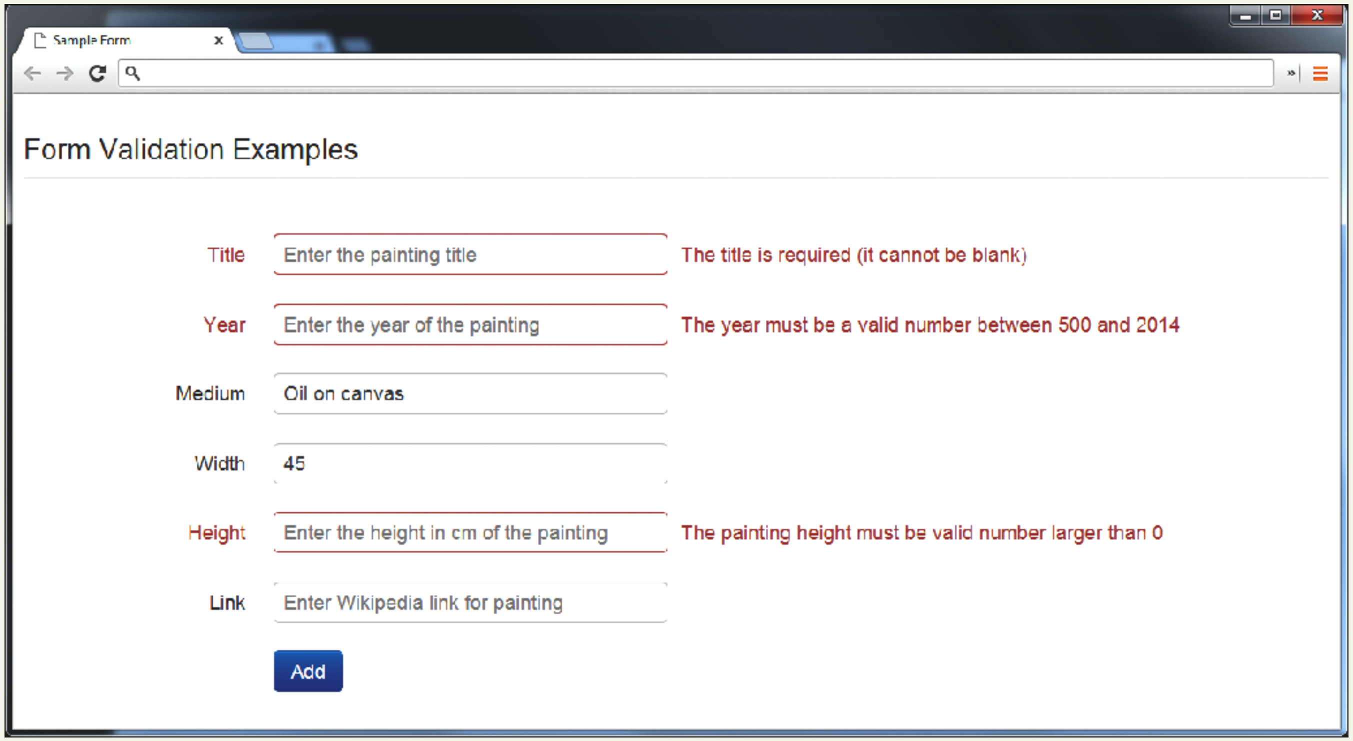
Notifying the User

We found an error, now what?

- **What is the problem?** Users do not want to read lengthy messages to determine what needs to be changed. They need to receive a visually clear and textually concise message.
- **Where is the problem?** Some type of error indication should be located near the field that generated the problem.
- **If appropriate, how do I fix it?** For instance, don’t just tell the user that a date is in the wrong format, tell him or her what format you are expecting, such as “The date should be in yy/mm/dd format.”

Another illustrative examples

What's wrong, where is it, and how to fix it.


Sample Form

Form Validation Examples

Title	Enter the painting title	The title is required (it cannot be blank)
Year	Enter the year of the painting	The year must be a valid number between 500 and 2014
Medium	Oil on canvas	
Width	45	
Height	Enter the height in cm of the painting	The painting height must be valid number larger than 0
Link	Enter Wikipedia link for painting	

Add

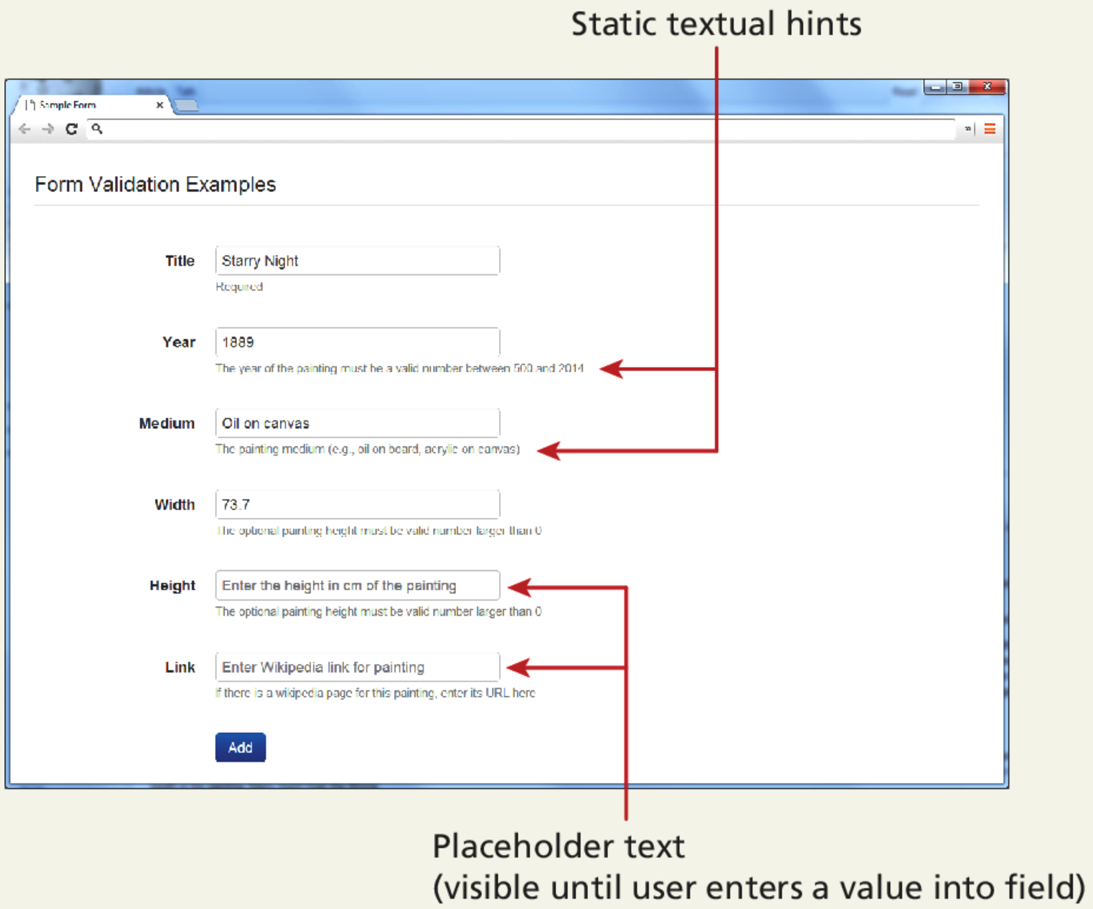
How to reduce validation errors

An ounce of prevention is worth a pound of cure

- Using pop-up JavaScript alert (or other popup) messages
- Provide textual hints to the user on the form itself
- Using tool tips or pop-overs to display context-sensitive help about the expected input
- a JavaScript-based mask

How to reduce validation errors

An ounce of prevention is worth a pound of cure



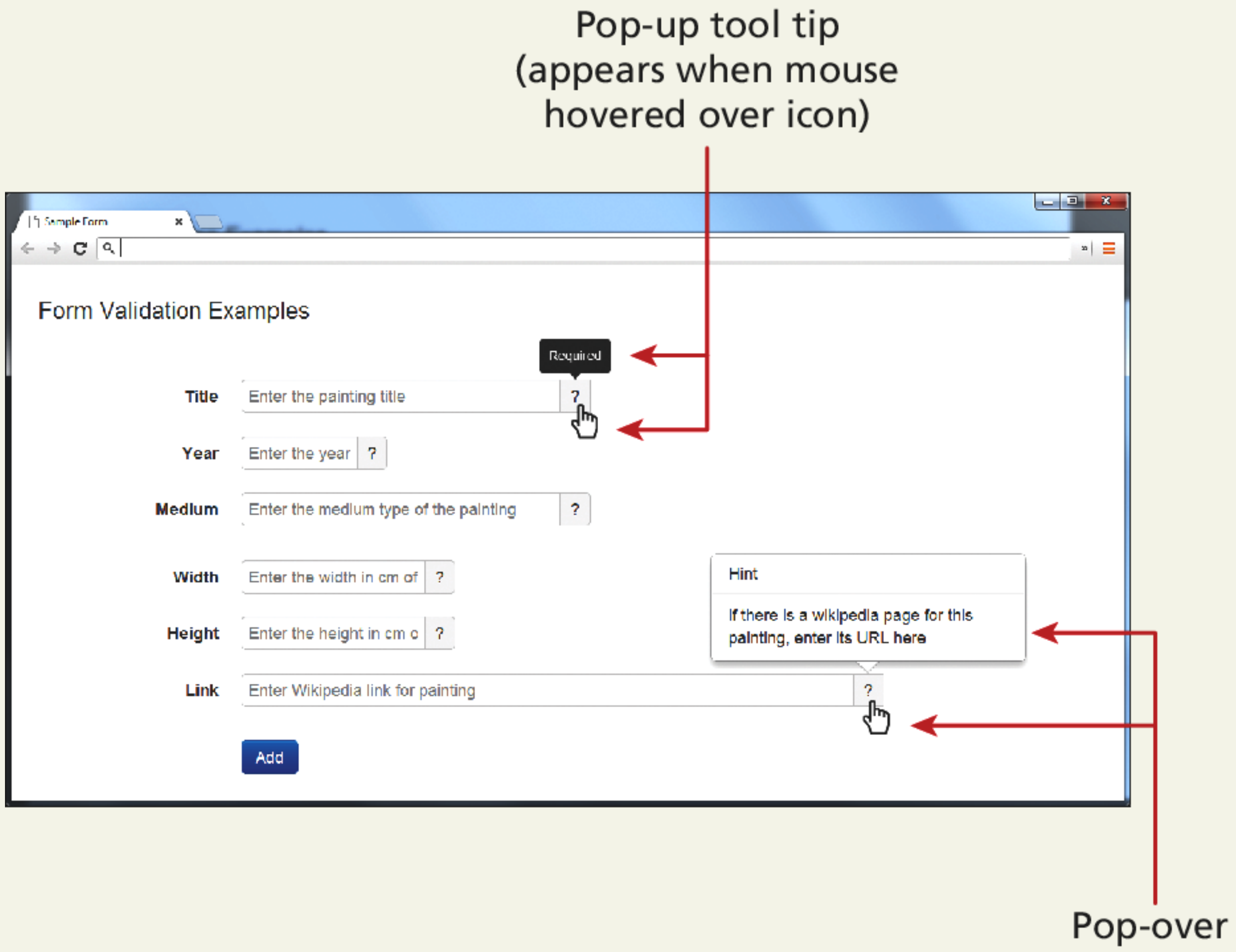
Form Validation Examples

Title	Starry Night
	Required
Year	1889
	The year of the painting must be a valid number between 500 and 2014
Medium	Oil on canvas
	The painting medium (e.g., oil on board, acrylic on canvas)
Width	73.7
	The optional painting height must be valid number larger than 0
Height	Enter the height in cm of the painting
	The optional painting height must be valid number larger than 0
Link	Enter Wikipedia link for painting
	If there is a wikipedia page for this painting, enter its URL here

```
<input type="text" … placeholder="Enter the height ...">
```

How to reduce validation errors

Tool Tips and popovers



Sample Form

Form Validation Examples

Required

Title	Enter the painting title ?
Year	Enter the year ?
Medium	Enter the medium type of the painting ?
Width	Enter the width in cm of ?
Height	Enter the height in cm o ?
Link	Enter Wikipedia link for painting ?

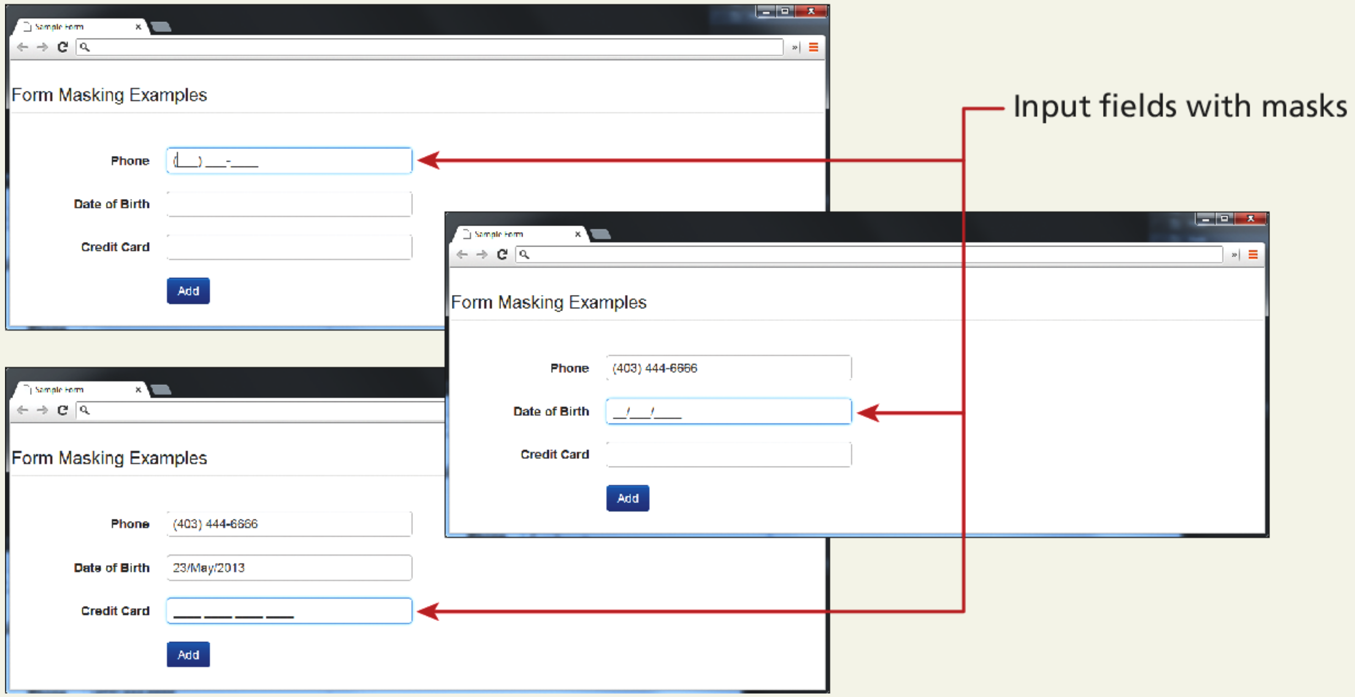
Hint

If there is a wikipedia page for this painting, enter its URL here

Add

How to reduce validation errors

JavaScript Mask



How to reduce validation errors

HTML 5 input types

Many user input errors can be eliminated by choosing a better data entry type than the standard

```
<input type="text">
```

If you need to get a date from the user, use the HTML5

```
<input type="date”>
```

If you need a number, use the HTML5

```
<input type="number">
```

CAPTCHA

Completely Automated Public Turing test to tell Computers and Humans Apart

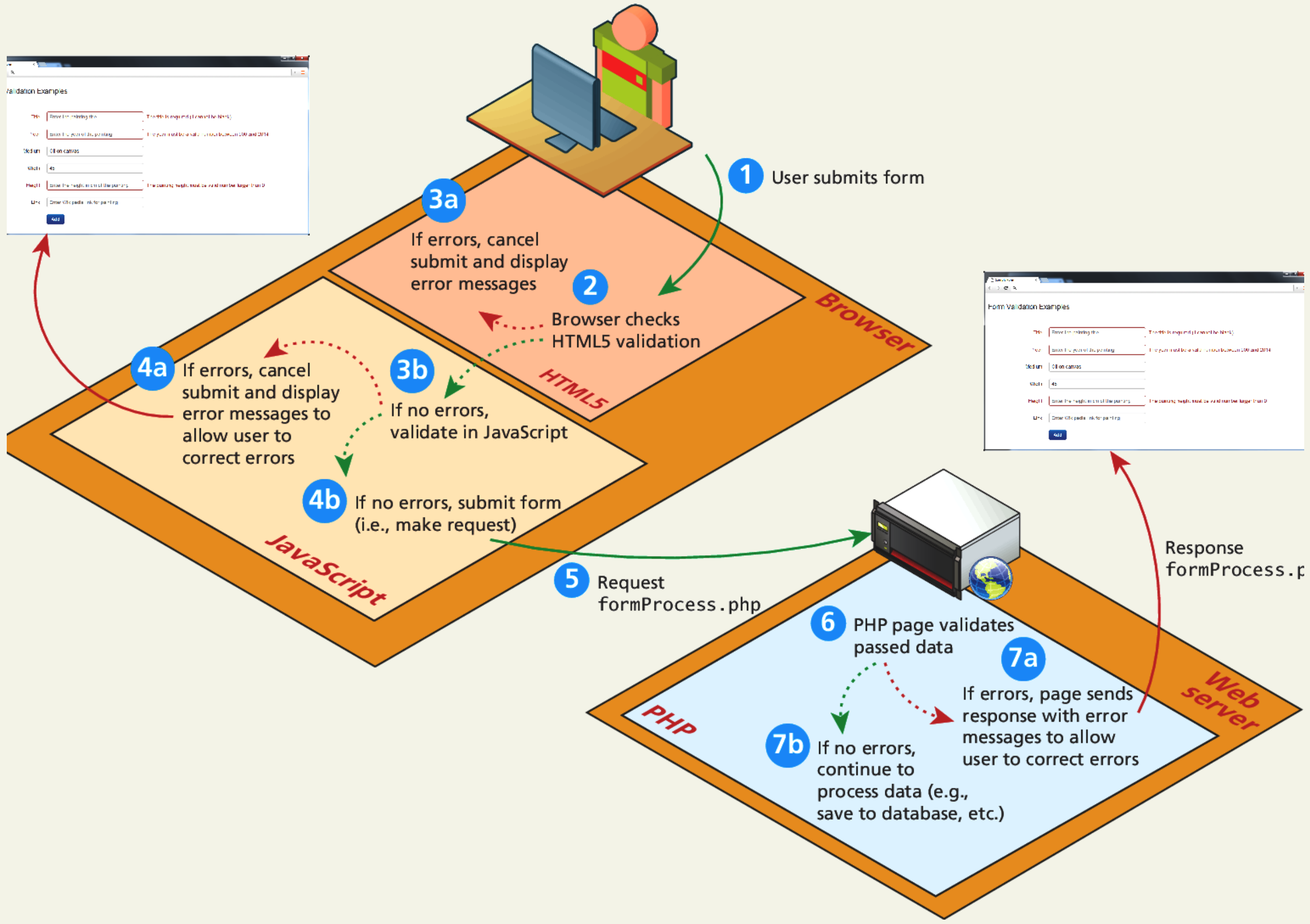
Automated form bots (often called **spam bots**) can flood a web application form with hundreds or thousands of bogus requests

This problem is generally solved by a test commonly referred to as a **CAPTCHA** which ask the user to enter a string of numbers and letters that are displayed in an obscured image that is difficult for a software bot to understand.

Section 6 of 6

WHERE TO PERFORM VALIDATION

Where to Validate?



1 User submits form
2 Browser checks HTML5 validation
3a If errors, cancel submit and display error messages
3b If no errors, validate in JavaScript
4a If errors, cancel submit and display error messages to allow user to correct errors
4b If no errors, submit form (i.e., make request)
5 Request formProcess.php
6 PHP page validates passed data
7a If errors, page sends response with error messages to allow user to correct errors
7b If no errors, continue to process data (e.g., save to database, etc.)
Response formProcess.p
Browser
HTML5
JavaScript
Web server
PHP
Validation Examples
Form Validation Examples

Where to Validate?

So many places

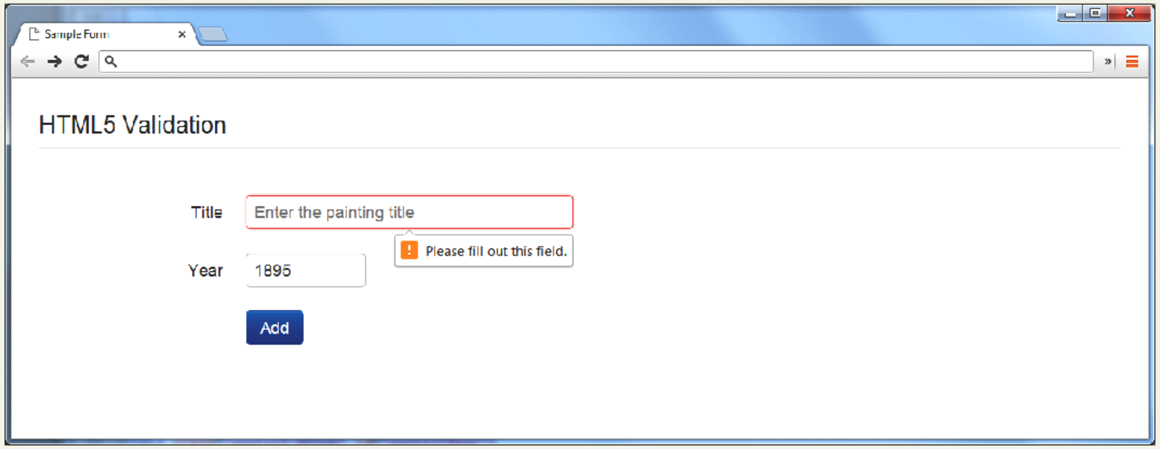
- Client-side using HTML5
- Client-Side using JavaScript
- **Server-Side using PHP**

While both client and server side validation is ideal, you must know that client-side scripts are not guaranteed to be executed. Therefore you must always perform server-side validation.

HTML5 validation

Client-Side

The *required* attribute can be added to an input element, and browsers that support it will perform their own validation and message.


HTML5 Validation

Title	Enter the painting title
Year	1895

Please fill out this field.

To disable HTML form validation

<form id="sampleForm" method=". . ." action=". . ." *novalidate*>

JavaScript validation

Client-Side

Consider that we want to validate on a form submit.

```
function init() {
    var sampleForm = document.getElementById('sampleForm');
    sampleForm.onsubmit = validateForm
;}
 // call the init function once all the html has been loaded
window.onload = init;
```

JavaScript validation

Client-Side

For instance, to check if the value in the form's password input element is between 8 and 16 characters, the JavaScript would be:

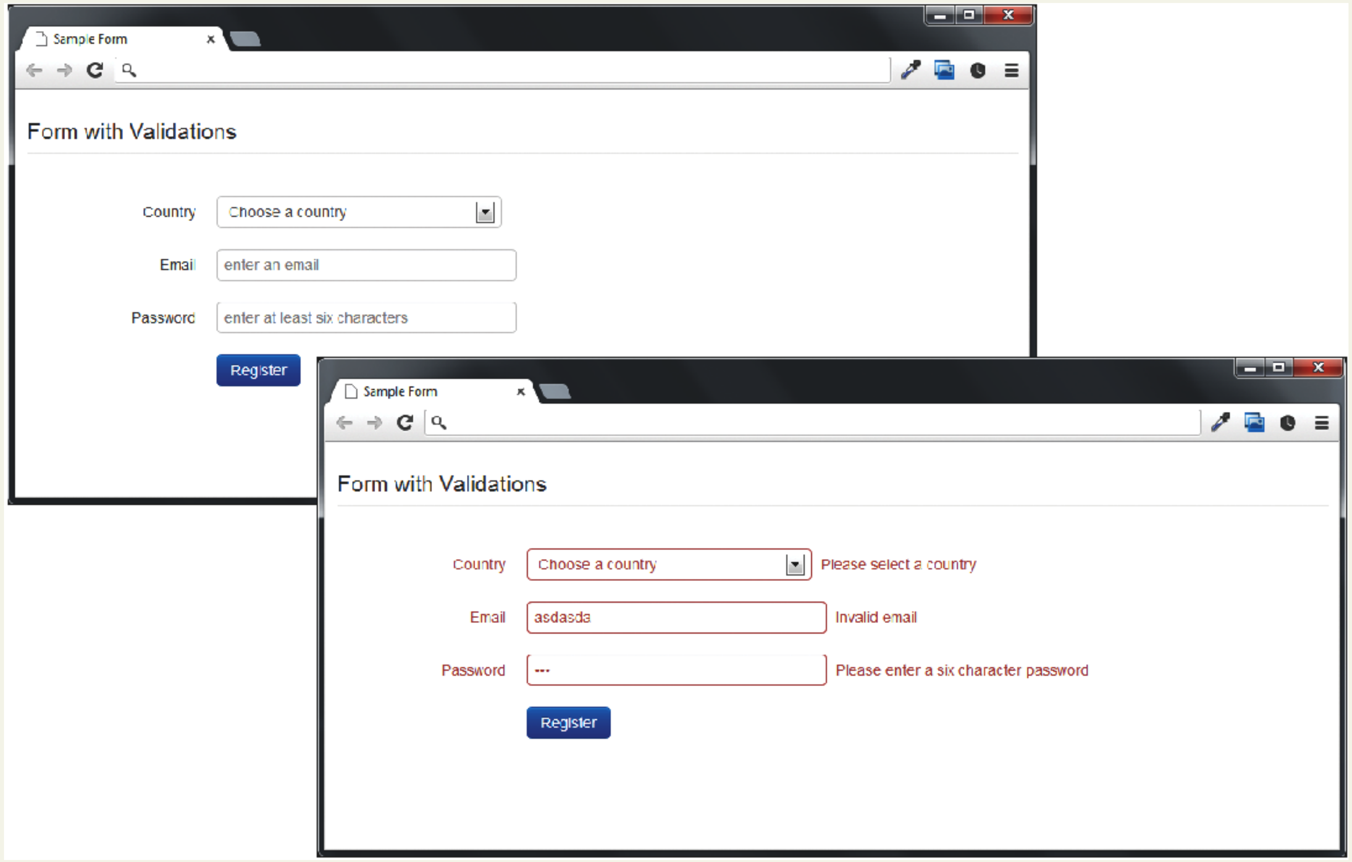
```
var passReg = /^[a-zA-Z]\w{8,16}$/;
if (! passReg.test(password.value)) {
    // provide some type of error message
}
```

What do we want to do when the JavaScript finds a validation error?

- Highlight errors by adding CSS classes to the input elements causing the error

JavaScript validation

Client-Side



Sample Form

Form with Validations

Country	Choose a country
Email	enter an email
Password	enter at least six characters

Register

Sample Form

Form with Validations

Country	Choose a country	Please select a country
Email	asdasda	Invalid email
Password	---	Please enter a six character password

Register

JavaScript Code

Function to add an error message to a certain elemenet (by id)

```
<script>
// we will reference these repeatedly
var country = document.getElementById('country');
var email = document.getElementById('email');
var password = document.getElementById('password');

/*
  Add passed message to the specified element
*/
function addErrorMessage(id, msg) {
   // get relevant span and div elements
   var spanId = 'error' + id;
   var span = document.getElementById(spanId);
   var divId = 'control' + id;
   var div = document.getElementById(divId);

   // add error message to error <span> element
   if (span) span.innerHTML = msg;
   // add error class to surrounding <div>
   if (div) div.className = div.className + " error";
}
```

JavaScript Code

Set up the event handlers

```
/*
  sets up event handlers
*/
function init() {
   var sampleForm = document.getElementById('sampleForm');
   sampleForm.onsubmit = validateForm;

   country.onchange = resetMessages;
   email.onchange = resetMessages;
   password.onchange = resetMessages;
}
```

JavaScript Code

The actual checks (part 1)

```
/*
  perform the validation checks
*/
function validateForm() {
   var errorFlag = false;

   // check email
   var emailReg = /(.+)@([^\.].*)\.([a-z]{2,})/;
   if (! emailReg.test(email.value)) {
      addErrorMessage('Email', 'Enter a valid email');
      errorFlag = true;
   }

   // check password
   var passReg = /^[a-zA-Z]\w{8,16}$/;
   if (! passReg.test(password.value)) {
      addErrorMessage('Password', 'Enter a password between 9-16
                       characters');
      errorFlag = true;
   }
```

JavaScript Code

The actual checks (part 2)

```
   // check country
   if ( country.selectedIndex <= 0 ) {
      addErrorMessage('Country', 'Select a country');
      errorFlag = true;
   }

   // if any error occurs then cancel submit; due to browser
   // irregularities this has to be done in a variety of ways
   if (! errorFlag)
      return true;
   else {
      if (e.preventDefault) {
         e.preventDefault();
      } else {
         e.returnValue = false;
      }
      return false;
   }
}

// set up validation handlers when page is downloaded and ready
window.onload = init;
```

LISTING 12.9 Complete JavaScript validation

PHP Validation

The only one you HAVE to do

No matter how good the HTML5 and JavaScript validation, client-side prevalidation can always be circumvented by hackers, or turned off by savvy users.

Validation on the server side using PHP is the most important form of validation and the only one that is absolutely essential.

PHP Validation

An abridged example…

```php
// if GET then just display form
//
// if POST then user has submitted data, we need to validate it
if ($_SERVER["REQUEST_METHOD"] == "POST") {
   $emailValid = ValidationResult::checkParameter("email",
                 '/(.+)@([^\.].*)\.([a-z]{2,})/',
                 'Enter a valid email [PHP]');
   $passValid = ValidationResult::checkParameter("password",
                '/^[a-zA-Z]\w{8,16}$/',
                'Enter a password between 8-16 characters [PHP]');
   $countryValid = ValidationResult::checkParameter("country",
                   '/[1-4]/', 'Choose a country [PHP]');

   // if no validation errors redirect to another page
   if ($emailValid->isValid() && $passValid->isValid() &&
                                 $countryValid->isValid() ) {
      header( 'Location: success.php' );
   }
```

PHP Va lidation

The only one you HAVE to do

```php
<?php
// turn on error reporting to help potential debugging
error_reporting(E_ALL);
ini_set('display_errors','1');

include_once('ValidationResult.class.php');

// create default validation results
$emailValid = new ValidationResult("", "", "", true);
$passValid = new ValidationResult("", "", "", true);
$countryValid = new ValidationResult("", "", "", true);

// if GET then just display form
//
// if POST then user has submitted data, we need to validate it
if ($_SERVER["REQUEST_METHOD"] == "POST") {
   $emailValid = ValidationResult::checkParameter("email",
                 '/(.+)@([^\.].*)\.([a-z]{2,})/',
                 'Enter a valid email [PHP]');
   $passValid = ValidationResult::checkParameter("password",
                '/^[a-zA-Z]\w{8,16}$/',
                'Enter a password between 8-16 characters [PHP]');
   $countryValid = ValidationResult::checkParameter("country",
                   '/[1-4]/', 'Choose a country [PHP]');

   // if no validation errors redirect to another page
   if ($emailValid->isValid() && $passValid->isValid() &&
                            $countryValid->isValid() ) {
      header( 'Location: success.php' );
   }
```

What You've Learned

1 What are Errors and Exceptions?

2 PHP Error Reporting

3 PHP Error and Exception Handling

4 Regular Expressions

5 Validating User Input

6 Where to Perform Validation