#Email

<form action="mailto:myaccount@mydomain.com">

<p>

<i>Please select the product you are interested in.</i>

</p>

<select name="product">

<option value="widget1">Widget #1</option>

<option value="widget2">Widget #2</option>

<option value="widget3">Widget #3</option>

</select>

<p>

<i>Please enter the text of your message in the field that follows.</i>

</p>

<textarea

name="body"

rows="10"

cols="60"></textarea>

<p>

<input type="submit" name="send" value="Send Your Message"/>

<input type="submit" name="cancel" value="Cancel - Never Mind"/>

</p>

</form>

This approach does give us a bit more control: the user is invited to select the product they are interested in, in addition to entering a message.

Unfortunately, there are several major problems with the mailto: form action strategy:

1. The user's browser might not be set up to send forms via email correctly.

2. The format in which the message actually reaches you varies from browser to browser, making it very difficult to read or organize the results.

3. **Internet Explorer displays a warning to the user** that email is about to be sent. Most users will just hit "Cancel" when they see warnings of this sort.

The mailto: approach might be handy for beginners who know just enough HTML to design a form. But for those of us who want to do professional work, there's a better way.

#### **Handling Email Forms With PHP**

"Real" form submissions— forms that transmit their data back to the web server, rather than directly sending email— are one of the most powerful features of HTML. By transmitting the user's message to the web server, we can control exactly how that message is handled— verifying that the user's entries are correct, requiring them to re-enter incorrect fields, and storing or emailing the final result in any way we see fit. Though this particular article focuses on email, we could just as easily choose to store the user's form submission in a database. Or store it in a database *and* email a notification to the webmaster. As the designer, the choice is up to you.

"Great! What HTML do I use to accomplish all this?"

Well, that's the catch: [HTML](https://www.boutell.com/newfaq/definitions/html.html) is a markup language, not a programming language. So if you have been searching for "the right HTML tags" to process a form submission, then your frustration is understandable.

Fortunately, you're definitely not the first person to want to make a website do more than deliver "static" HTML pages! So [web servers](https://www.boutell.com/newfaq/definitions/server.html) have long since been extended in a variety of ways to allow "dynamic" behavior such as handling form submissions. Currently the most popular solution is [PHP](https://www.boutell.com/newfaq/definitions/php.html).

PHP is a simple programming language that can easily be "sprinkled into" a web page to add new capabilities like handling form submissions. Unlike earlier approaches, such as CGI (Common Gateway Interface) programming, PHP puts the web page first and lets you add a little bit of PHP code at a time as your needs grow more complex. And that makes PHP easier to learn.

#### **Learning PHP**

PHP is too big a subject for a single article. And that's why I've written a separate article on it. See my article [how can I receive form submissions?](https://www.boutell.com/newfaq/creating/forms.html) for a whirlwind introduction to PHP. That article also links to several more detailed tutorials that teach PHP programming in greater depth.

The rest of this article will focus on an example of a useful PHP page that allows the user to send email to the webmaster. We'll take advantage of the CAPTCHA system presented in my article [how do I add a CAPTCHA to my web page?](https://www.boutell.com/newfaq/creating/captcha.html) to make life difficult for spammers. And we'll use the Accountify system introduced in my article [how do I add accounts to my website?](https://www.boutell.com/newfaq/creating/accounts.html) to pre-fill the "email address" and "real name" blanks on the form and skip the CAPTCHA for users who have logged into accounts on our website. Both of these features are optional and you can use the form without them. In particular, websites that have no other use for accounts probably won't want to use Accountify just for a single page.

#### **The Basics: A Bare-Bones Email Form With PHP**

Here's a dead-simple web page, myform.html, that submits form data to a PHP page called takeform.php:

<html>

<head>

<title>Send Us Email</title>

</head>

<body>

<form action="takeform.php" method="POST">

<p>

<input

name="email"

size="64"

maxlength="64"/>

<b>Your</b> Email Address

</p>

<p>

<input

name="realname"

size="64"

maxlength="64"/>

Your Real Name (<i>so our reply won't get stuck in your spam folder</i>)

</p>

<p>

<input

name="subject"

size="64"

maxlength="64"/>

Subject Of Your Message

</p>

<p>

<i>Please enter the text of your message in the field that follows.</i>

</p>

<textarea

name="body"

rows="10"

cols="60">

</textarea>

<p>

<input type="submit" name="send" value="Send Your Message"/>

</p>

</form>

</body>

</html>

And here is the code for takeform.php, a PHP page that accepts the form submission and delivers it to the webmaster:

<html>

<head>

<title>Thanks For Contacting Us</title>

</head>

<body>

<?php

// Change this to YOUR address

$recipient = 'webmaster@example.com';

$email = $\_POST['email'];

$realName = $\_POST['realname'];

$subject = $\_POST['subject'];

$body = $\_POST['body'];

# We'll make a list of error messages in an array

$messages = array();

# Allow only reasonable email addresses

if (!preg\_match("/^[\w\+\-.~]+\@[\-\w\.\!]+$/", $email)) {

$messages[] = "That is not a valid email address.";

}

# Allow only reasonable real names

if (!preg\_match("/^[\w\ \+\-\'\"]+$/", $realName)) {

$messages[] = "The real name field must contain only " .

"alphabetical characters, numbers, spaces, and " .

"reasonable punctuation. We apologize for any inconvenience.";

}

# CAREFUL: don't allow hackers to sneak line breaks and additional

# headers into the message and trick us into spamming for them!

$subject = preg\_replace('/\s+/', ' ', $subject);

# Make sure the subject isn't blank afterwards!

if (preg\_match('/^\s\*$/', $subject)) {

$messages[] = "Please specify a subject for your message.";

}

$body = $\_POST['body'];

# Make sure the message has a body

if (preg\_match('/^\s\*$/', $body)) {

$messages[] = "Your message was blank. Did you mean to say " .

"something?";

}

if (count($messages)) {

# There were problems, so tell the user and

# don't send the message yet

foreach ($messages as $message) {

echo("<p>$message</p>\n");

}

echo("<p>Click the back button and correct the problems. " .

"Then click Send Your Message again.</p>");

} else {

# Send the email - we're done

mail($recipient,

$subject,

$body,

"From: $realName <$email>\r\n" .

"Reply-To: $realName <$email>\r\n");

echo("<p>Your message has been sent. Thank you!</p>\n");

}

?>

</body>

</html>

#php code for email

<?php

if(isset($\_POST['email'])) {

// EDIT THE 2 LINES BELOW AS REQUIRED

$email\_to = "you@yourdomain.com";

$email\_subject = "Your email subject line";

function died($error) {

// your error code can go here

echo "We are very sorry, but there were error(s) found with the form you submitted. ";

echo "These errors appear below.<br /><br />";

echo $error."<br /><br />";

echo "Please go back and fix these errors.<br /><br />";

die();

}

// validation expected data exists

if(!isset($\_POST['first\_name']) ||

!isset($\_POST['last\_name']) ||

!isset($\_POST['email']) ||

!isset($\_POST['telephone']) ||

!isset($\_POST['comments'])) {

died('We are sorry, but there appears to be a problem with the form you submitted.');

}

$first\_name = $\_POST['first\_name']; // required

$last\_name = $\_POST['last\_name']; // required

$email\_from = $\_POST['email']; // required

$telephone = $\_POST['telephone']; // not required

$comments = $\_POST['comments']; // required

$error\_message = "";

$email\_exp = '/^[A-Za-z0-9.\_%-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}$/';

if(!preg\_match($email\_exp,$email\_from)) {

$error\_message .= 'The Email Address you entered does not appear to be valid.<br />';

}

$string\_exp = "/^[A-Za-z .'-]+$/";

if(!preg\_match($string\_exp,$first\_name)) {

$error\_message .= 'The First Name you entered does not appear to be valid.<br />';

}

if(!preg\_match($string\_exp,$last\_name)) {

$error\_message .= 'The Last Name you entered does not appear to be valid.<br />';

}

if(strlen($comments) < 2) {

$error\_message .= 'The Comments you entered do not appear to be valid.<br />';

}

if(strlen($error\_message) > 0) {

died($error\_message);

}

$email\_message = "Form details below.\n\n";

function clean\_string($string) {

$bad = array("content-type","bcc:","to:","cc:","href");

return str\_replace($bad,"",$string);

}

$email\_message .= "First Name: ".clean\_string($first\_name)."\n";

$email\_message .= "Last Name: ".clean\_string($last\_name)."\n";

$email\_message .= "Email: ".clean\_string($email\_from)."\n";

$email\_message .= "Telephone: ".clean\_string($telephone)."\n";

$email\_message .= "Comments: ".clean\_string($comments)."\n";

// create email headers

$headers = 'From: '.$email\_from."\r\n".

'Reply-To: '.$email\_from."\r\n" .

'X-Mailer: PHP/' . phpversion();

@mail($email\_to, $email\_subject, $email\_message, $headers);

?>

<!-- include your own success html here -->

Thank you for contacting us. We will be in touch with you very soon.

<?php

}

?>

#2

<?php  
if($\_POST["submit"]) {  
 $recipient="yourmailid@domain.com"; //Enter your mail address  
 $subject="Contact from Website"; //Subject   
 $sender=$\_POST["name"];  
 $senderEmail=$\_POST["email"];  
 $message=$\_POST["comments"];  
 $mailBody="Name: $sender\nEmail Address: $senderEmail\n\nMessage: $message";  
 mail($recipient, $subject, $mailBody);  
 sleep(1);  
 header("Location:http://blog.antonyraphel.in/sample/"); // Set here redirect page or destination page  
}  
?>

#3

<form method="post" name="contact\_form"  
action="contact-form-handler.php">  
 Your Name:  
 <input type="text" name="name">  
 Email Address:  
 <input type="text" name="email">  
 Message:  
 <textarea name="message"></textarea>  
 <input type="submit" value="Submit">  
</form>

JS:

<script language="JavaScript">  
var frmvalidator = new Validator("contactform");  
frmvalidator.addValidation("name","req","Please provide your name");  
frmvalidator.addValidation("email","req","Please provide your email");  
frmvalidator.addValidation("email","email",  
 "Please enter a valid email address");  
</script>

PHP:

<?php  
$errors = '';  
$myemail = 'yourname@website.com';//<-----Put Your email address here.  
if(empty($\_POST['name']) ||  
 empty($\_POST['email']) ||  
 empty($\_POST['message']))  
{  
 $errors .= "\n Error: all fields are required";  
}  
$name = $\_POST['name'];  
$email\_address = $\_POST['email'];  
$message = $\_POST['message'];  
if (!preg\_match(  
"/^[\_a-z0-9-]+(\.[\_a-z0-9-]+)\*@[a-z0-9-]+(\.[a-z0-9-]+)\*(\.[a-z]{2,3})$/i",  
$email\_address))  
{  
 $errors .= "\n Error: Invalid email address";  
}  
  
if( empty($errors))  
{  
$to = $myemail;  
$email\_subject = "Contact form submission: $name";  
$email\_body = "You have received a new message. ".  
" Here are the details:\n Name: $name \n ".  
"Email: $email\_address\n Message \n $message";  
$headers = "From: $myemail\n";  
$headers .= "Reply-To: $email\_address";  
mail($to,$email\_subject,$email\_body,$headers);  
//redirect to the 'thank you' page  
header('Location: contact-form-thank-you.html');  
}  
?>

#session handling

session\_start();  
  
  
$\_SESSION['name'] = $\_POST['os0'];  
$\_SESSION['number'] = $\_POST['os1'];  
$\_SESSION['email'] = $\_POST['os2'];

Page 3

session\_start();  
  
  
$\_SESSION['collection\_date'] = $\_POST['os4'];  
$\_SESSION['address'] = $\_POST['address'];

Page 4

session\_start();  
  
$\_SESSION['packkit'] = $\_POST['os5'];  
$\_SESSION['delivery'] = $\_POST['on6'];  
$\_SESSION['additems'] = $\_POST['os7'];

php to insert into database

<?php  
session\_start();  
  
$con = mysql\_connect("XXXX","XXXXX","XXXXX");  
**if** (!$con)  
 {  
 **die**('Could not connect: ' . mysql\_error());  
 }  
  
mysql\_select\_db("XXXXX", $con);  
  
$number = mysql\_real\_escape\_string(preg\_replace('/[^0-9]/', '', $\_SESSION['number']));  
  
$realname = mysql\_real\_escape\_string($\_SESSION['name']);  
  
$email = mysql\_real\_escape\_string($\_SESSION['email']);  
  
$address = mysql\_real\_escape\_string(preg\_replace('/[\r\n]/',' ',$\_SESSION['address']));  
  
$collection = mysql\_real\_escape\_string($\_SESSION['collection\_date']);  
  
$packingkit = mysql\_real\_escape\_string($\_SESSION['packkit']);  
  
$delivery = mysql\_real\_escape\_string($\_SESSION['delivery']);  
  
$additems = mysql\_real\_escape\_string($\_SESSION['additems']);  
  
$tandc = mysql\_real\_escape\_string($\_POST['tandc']);  
  
$sql="INSERT INTO Orders (Name, Number, Email, Size, Address, Collection\_Date, Packing\_kit, Delivery\_date, Additional\_Items, Date\_received, Terms, Deposit)  
VALUES  
('$realname','$number','$email','Standard','$address','$collection','$packingkit','$delivery','$additems', date\_format(curdate(),'%d-%m-%Y'),'$tandc', 'Yes')";  
  
**if** (!mysql\_query($sql,$con))  
{  
**die**('Error: ' . mysql\_error());  
}  
**echo** "";  
  
mysql\_close($con)  
?>