

Chapter 22

How to send email and access other web sites

Objectives

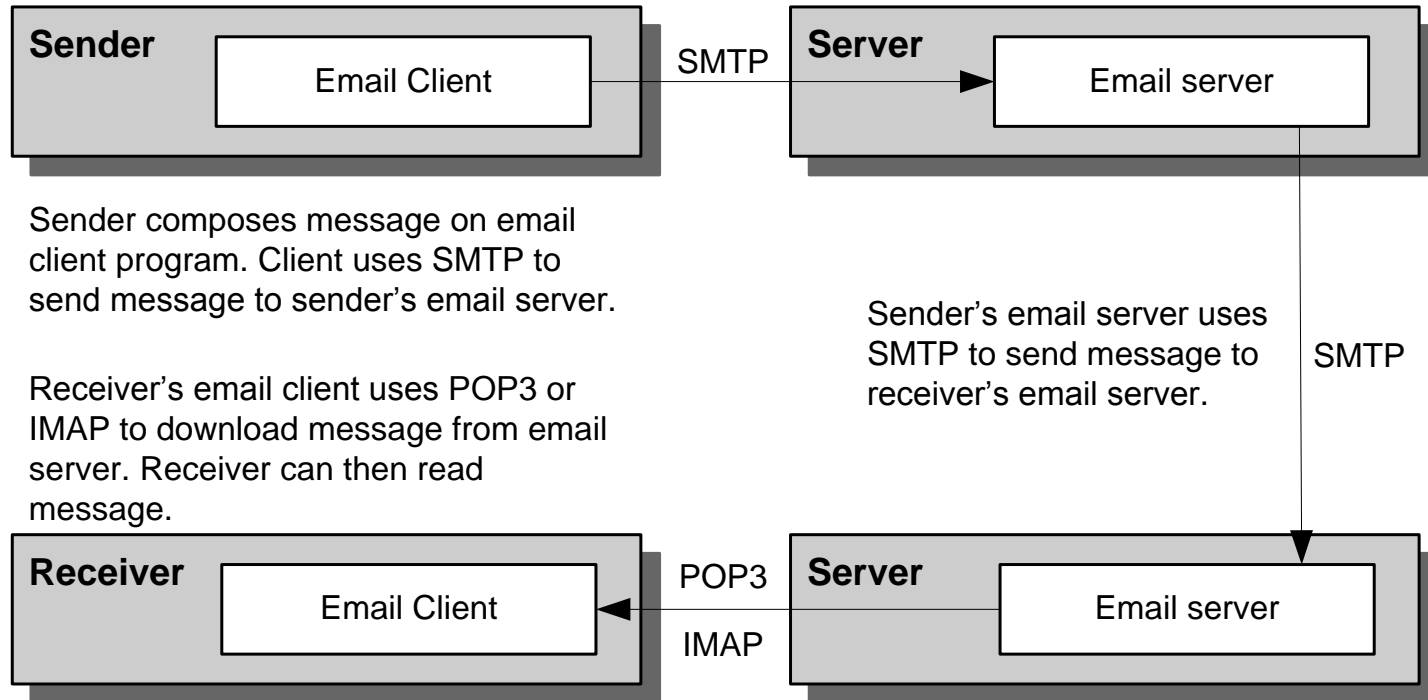
Applied

1. Install and use the PEAR Mail package to send email messages from your web applications.
2. Use the cURL library to work with the data on web sites that provide APIs for doing that.

Knowledge

1. Describe the use of the PEAR Mail package, including its static factory method and its mailer object.
2. Describe the use of a helper function for sending email with the PEAR Mail package.
3. Describe the use of the cURL library for working with the data on a web site that provides an API for doing that.

How email is sent



Protocols used in sending email

- SMTP
- POP3
- IMAP

Terms

- email client
- email server

Limitations of the built-in PHP mail function

- You must modify the php.ini file to change email servers.
- You cannot use an encrypted connection to the email server.
- You cannot provide a username and password when connecting to the email server.

How to install the PEAR Mail package in Windows

1. Click the Start menu, select the Programs→Accessories group, and click on Command Prompt to open a command prompt window.
2. In the Command Prompt window, type these commands:

```
cd \xampp\php  
pear install --alldeps Mail  
exit
```

How to install the PEAR Mail package in Mac OS X

1. In Finder, open the Applications folder, open the Utilities folder, and double click the Terminal icon to open a Terminal window.
2. In the Terminal window, type these commands:

```
cd /Applications/XAMPP/xamppfiles/bin  
sudo ./pear install --alldeps Mail  
exit
```

How to create a Gmail account for email testing

1. If you don't already have a Gmail account, create one. To do that, you can go to www.gmail.com, click on the “Create an Account” link, and follow the instructions.
2. Login to your Gmail account and go to your Inbox.
3. Click the “Settings” link at the top of the page.
4. On the Settings page, click the “Forwarding and POP/IMAP” link.
5. In the POP download section, select the “Enable POP for all mail” option.
6. Click the “Save Changes” button.

Configuration info for a Gmail SMTP server

Setting	Value
Host name	smtp.gmail.com
Encryption	SSL
Port number	465
Authentication	Yes
Username	<i>username@gmail.com</i>
Password	<i>yourGmailPassword</i>

Step 1: Load the PEAR Mail package

```
require_once 'Mail.php';
```

Step 2: Set the parameters for the mailer object

Example 1: A simple SMTP server

```
$options = array();  
$options['host'] = 'mail.example.com';
```

Example 2: An SMTP server with authentication

```
$options = array();  
$options['host'] = 'mail.example.com';  
$options['auth'] = true;  
$options['username'] = 'sample.user@example.com';  
$options['password'] = 'Sup3r*S3cr3+';
```

Example 3: SMTP server with authentication and SSL encryption

```
$options = array();  
$options['host'] = 'ssl://mail.example.com';  
$options['port'] = 465;  
$options['auth'] = true;  
$options['username'] = 'mail.account@example.com';  
$options['password'] = 'Sup3r*S3cr3+';
```

Step 3: Create the mailer object

```
$mailer = Mail::factory('smtp', $options);
```

Step 4: Send the message

```
// 1. Set the SMTP headers
$headers = array();
$headers['From'] = 'sample.user@example.com';
$headers['To'] = 'john.doe@example.com,
jane.doe@example.com';
$headers['Cc'] = 'fsmith@example.org';
$headers['Bcc'] = 'jsmith@example.net';
$headers['Subject'] = 'How to use PEAR Mail';

// 2. Set the recipient list
$recipients = 'john.doe@example.com,
jane.doe@example.com, ';
$recipients .= 'fsmith@example.org, jsmith@example.net';

// 3. Set the text for the body of the message
$body =
    "The Murach PHP and MySQL book has a chapter on\n";
$body .=
    "how to use the PEAR Mail package to send email.";
```

Step 4: Send the message (continued)

```
// 4. Send the message
$result = $mailer->send($recipients, $headers, $body);

// 5. Check the result and display an error if one exists
if (PEAR::isError($result)) {
    $error =
        'Error sending email: ' . $result->getMessage();
    echo htmlspecialchars($error);
}
```

How to use HTML formatting in the body of the message

```
// 1. Set the Content-type header
$headers['Content-type'] = 'text/html';

// 2. Use HTML tags to format the body of the message
$body = '<p>The Murach PHP and MySQL book has a chapter
        on how to use the
        <a href="http://pear.php.net/Mail/">
            PEAR Mail package
        </a>
        to send email.
    </p>';
```

The message.php file (a helper function)

```
<?php
require_once 'Mail.php';
require_once 'Mail/RFC822.php';

function send_email($to, $from, $subject, $body,
$is_body_html = false) {
    if (! valid_email($to)) {
        throw new Exception('This To address is invalid: ' .
            htmlspecialchars($to));
    }
    if (! valid_email($from)) {
        throw new Exception('This From address is invalid: ' .
            htmlspecialchars($from));
    }
}
```

The message.php file (continued)

```
$smtp = array();  
// **** You must change the following to match your  
// **** SMTP server and account information.  
$smtp['host'] = 'ssl://smtp.gmail.com';  
$smtp['port'] = 465;  
$smtp['auth'] = true;  
$smtp['username'] = 'example@gmail.com';  
$smtp['password'] = 'supersecret';  
  
$mailer = Mail::factory('smtp', $smtp);  
if (PEAR::isError($mailer)) {  
    throw new Exception('Could not create mailer.');}  
  
// Add the email address to an array of all recipients  
$recipients = array();  
$recipients[] = $to;
```


The message.php file (continued)

```
// Set the headers
$headers = array();
$headers['From'] = $from;
$headers['To']   = $to;
$headers['Subject'] = $subject;
if ($is_body_html) {
    $headers['Content-type'] = 'text/html';
}

// Send the email
$result = $mailer->send($recipients, $headers, $body);

// Check the result and throw an error if one exists
if (PEAR::isError($result)) {
    throw new Exception('Error sending email: ' .
        htmlspecialchars($result->getMessage()) );
}
}
```

The message.php file (continued)

```
function valid_email($email) {  
    $emailObjects = Mail_RFC822::parseAddressList($email);  
    if (PEAR::isError($emailObjects)) return false;  
  
    // Get mailbox and host parts of first email object  
    $mailbox = $emailObjects[0]->mailbox;  
    $host = $emailObjects[0]->host;  
  
    // Make sure mailbox and host parts aren't too long  
    if (strlen($mailbox) > 64) return false;  
    if (strlen($host) > 255) return false;
```

The message.php file (continued)

```
// Validate the mailbox
$atom = '[:alnum:]_!#$%&\'+\./=?^`{|}~-' + ' ';
$dotatom = '(\.' . $atom . ')*';
$address = ' (^' . $atom . $dotatom . '$) ';
$char = '([^\\"'])';
$esc = '(\\"')';
$text = '(' . $char . '|' . $esc . ')+';
$quoted = ' (^"' . $text . '"$) ';
$localPart = '/' . $address . '|' . $quoted . '/';
$localMatch = preg_match($localPart, $mailbox);
if ($localMatch === false || $localMatch != 1) {
    return false; }
```

The message.php file (continued)

```
// Validate the host
$hostname =
    '([[:alnum:]]([-[:alnum:]]{0,62}[[:alnum:]])?)';
$hostnames =
    '(' . $hostname . '(\.' . $hostname . ')*)';
$stop = '\.[[:alnum:]]{2,6}';
$domainPart = '/^' . $hostnames . $stop . '$/';
$domainMatch = preg_match($domainPart, $host);
if ($domainMatch === false || $domainMatch != 1) {
    return false; }

return true;
}
?>
```

How to use the send_email function

```
require_once 'message.php';

$from = 'John Doe <john.doe@example.com>';
$to = 'Jane Doe <jane.doe@example.com>';

$subject = 'How to use PEAR Mail';

$body =
    '<p>The Murach PHP and MySQL book has a chapter on how
      to use the
      <a href="http://pear.php.net/Mail/">
        PEAR Mail package</a>
      to send email.</p>';

$is_body_html = true;
try {
    send_email($to, $from, $subject, $body, $is_body_html);
} catch (Exception $e) {
    $error = $e->getMessage();
    echo $error;
}
```

How to configure PHP to use the cURL library in Windows

1. Open the php.ini file in a text editor.
2. Find the line that contains the text “php_curl.dll”.
3. Remove the semicolon from the beginning of the line. It should read as follows:
extension=php_curl.dll
4. Restart the Apache web server as described in the appendix.

How to configure PHP to use the cURL library in Mac OS X

1. Open the php.ini file in a text editor.
2. Find the line that contains the text “curl.so”.
3. If necessary, remove the semicolon from the beginning of the line.
It should read as follows:
extension=curl.so
4. Restart the Apache web server as described in the appendix.

Common cURL functions

```
curl_init($url)
```

```
curl_setopt($curl, OPTION, $value)
```

```
curl_exec($curl)
```

```
curl_close($curl)
```


How to use the cURL functions

```
// Initialize the cURL session
$curl = curl_init('http://www.example.com');

// Set the cURL options so the session returns data
curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);

// Transfer the data and store it
$page = curl_exec($curl);

// Close the session
curl_close($curl);

// Process the data
$page = nl2br(htmlspecialchars($page));
echo $page;
```

The URL for the YouTube API documentation

<http://code.google.com/apis/youtube/2.0/reference.html>

The base URL for the YouTube Video Search API

<http://gdata.youtube.com/feeds/api/videos>

Four parameters that you can add to the base URL

alt	Specifies an alternate data format. The default is “atom” which returns XML data, but you can use “json” to return data in the JavaScript Object Notation (JSON) format.
q	Specifies the search string.
orderby	Specifies how to sort the results of the search. The default is “relevance”; you can specify other values such as “published”, “viewCount”, and “rating”.
safeSearch	Filters videos by removing restricted content. The default value is “moderate”; you can specify other values such as “none” and “strict”.

A URL for a YouTube query

`http://gdata.youtube.com/feeds/api/videos?alt=json&q=shuttle`

How to use cURL to query YouTube

```
// Set up the URL for the query
$query = 'space shuttle';
$query = urlencode($query);
$base_url = 'http://gdata.youtube.com/feeds/api/videos';
$params = 'alt=json&q=' . $query;
$url = $base_url . '?' . $params;

// Use cURL to get data in JSON format
$curl = curl_init($url);
curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
$json_data = curl_exec($curl);
curl_close($curl);

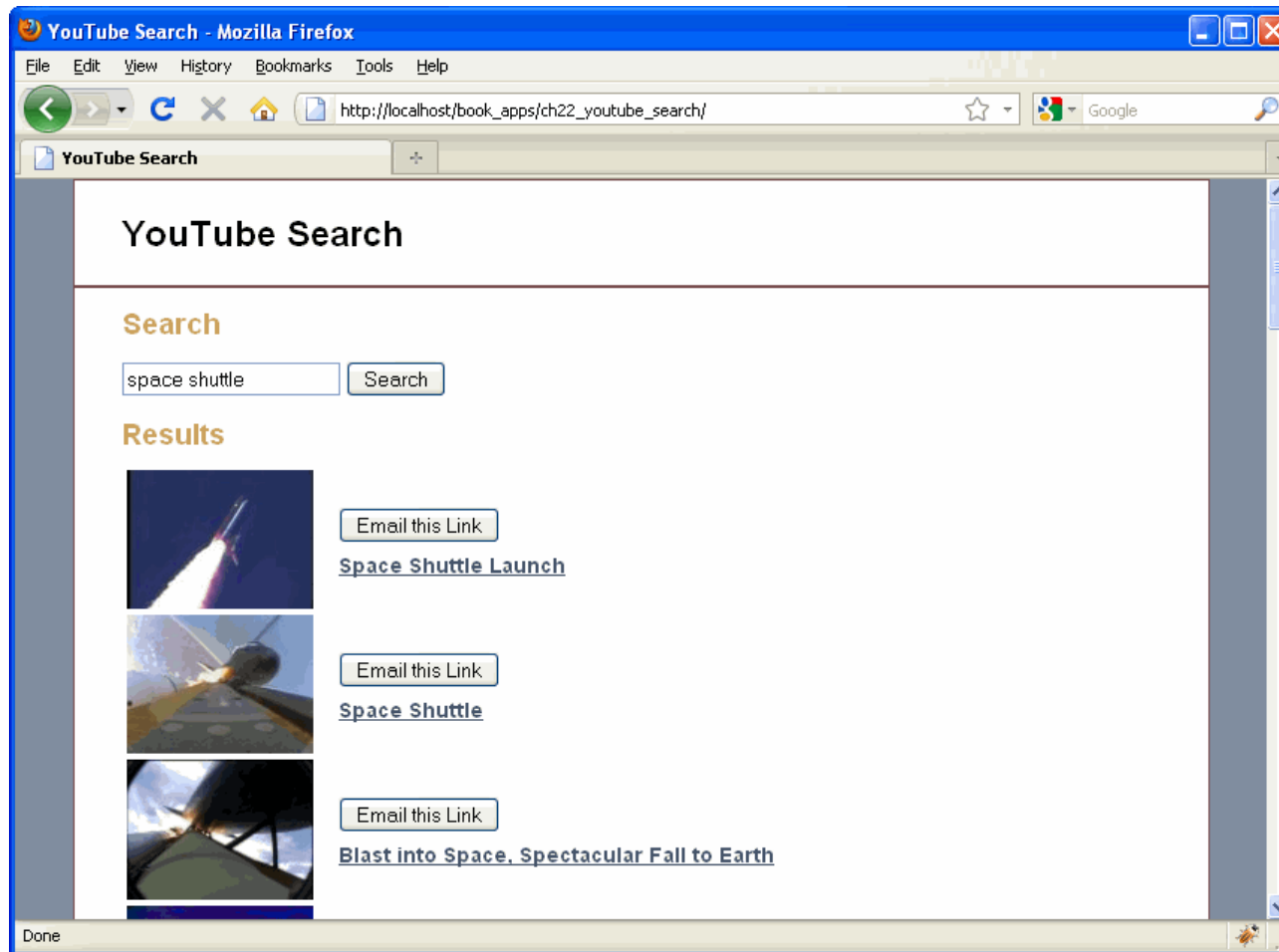
// Get an array of videos from the JSON data
$data = json_decode($json_data, true);
$videos = $data['feed']['entry'];
```

How to use cURL to query YouTube (continued)

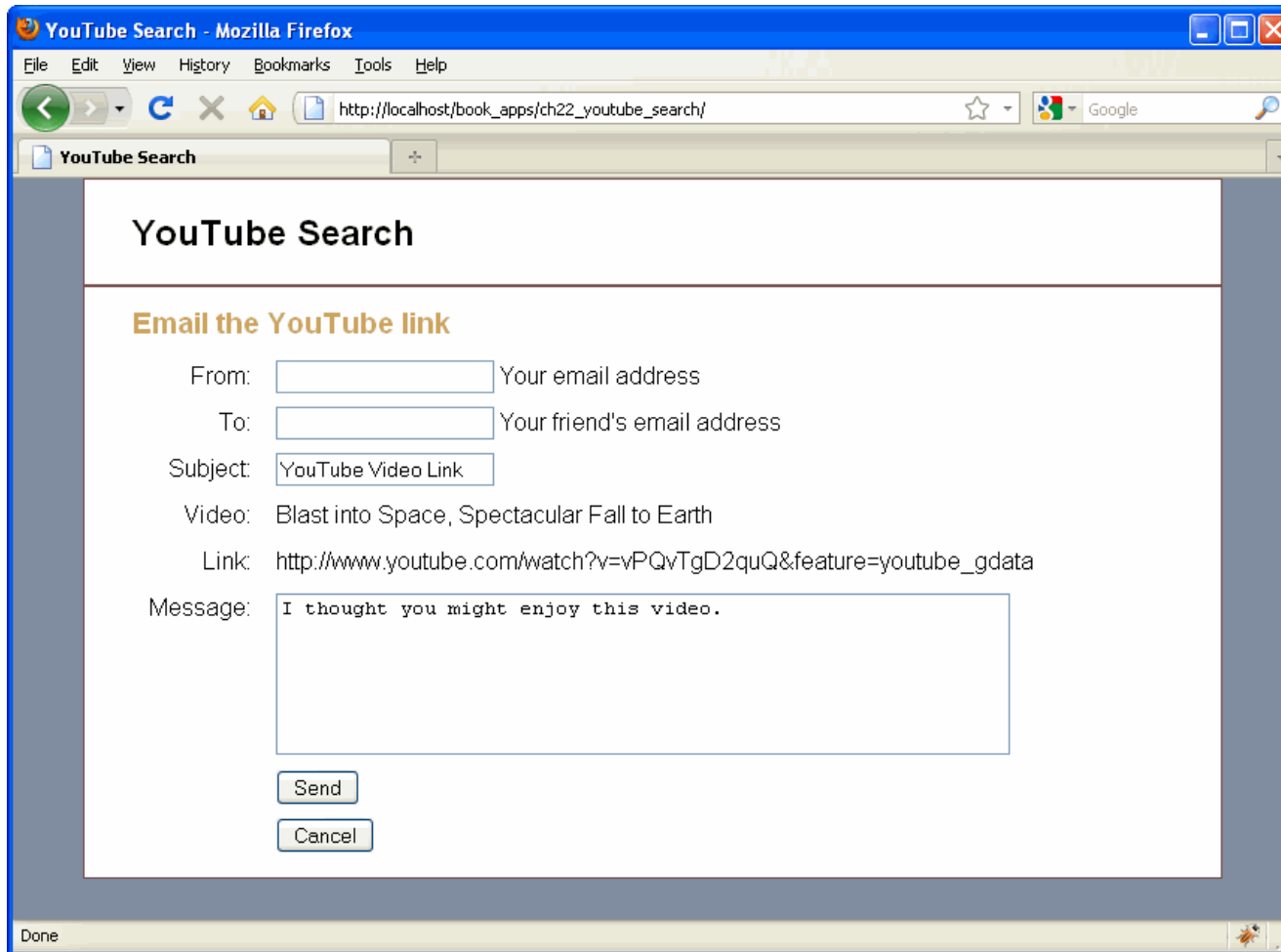
```
// Access the data for each video
foreach ($videos as $video) {
    $image_url =
        $video['media$group']['media$thumbnail'][0]['url'];
    $video_url = $video['link'][0]['href'];
    $text      = $video['title']['$t'];

    // Code to output these values
}
```

Search view



Email view



The screenshot shows a Mozilla Firefox browser window titled "YouTube Search". The address bar displays the URL "http://localhost/book_apps/ch22_youtube_search/". The page content includes a header "YouTube Search" and a section titled "Email the YouTube link". Below this section is an email form with the following fields and values:

- From: Your email address
- To: Your friend's email address
- Subject:
- Video: Blast into Space, Spectacular Fall to Earth
- Link: http://www.youtube.com/watch?v=vPQvTgD2quQ&feature=youtube_gdata
- Message:

I thought you might enjoy this video.

At the bottom of the form are two buttons: "Send" and "Cancel". The browser's status bar at the bottom left shows "Done".

The controller (index.php)

```
<?php
require_once 'message.php';
session_start();

function search_youtube($query) {
    // Set up the URL for the query
    $query = urlencode($query);
    $base_url = 'http://gdata.youtube.com/feeds/api/videos';
    $params = 'alt=json&q=' . $query;
    $url = $base_url . '?' . $params;

    // Use cURL to get data in JSON format
    $curl = curl_init($url);
    curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
    $json_data = curl_exec($curl);
    curl_close($curl);

    // Get array of videos from the JSON data and return it
    $data = json_decode($json_data, true);
    $videos = $data['feed']['entry'];
    return $videos; }
```


The controller (continued)

```
if (isset($_POST['action'])) {  
    $action = $_POST['action'];  
} else {  
    $action = 'search';  
}  
  
if (isset($_POST['query'])) {  
    $query = $_POST['query'];  
    $_SESSION['query'] = $query;  
} else if (isset($_SESSION['query'])) {  
    $query = $_SESSION['query'];  
} else {  
    $query = '';  
}
```

The controller (continued)

```
switch ($action) {  
    case 'search':  
        if (!empty($query)) {  
            $videos = search_youtube($query);  
        }  
        include 'search_view.php';  
        break;  
    case 'display_email_view':  
        $url = $_POST['url'];  
        $text = $_POST['text'];  
        include 'email_view.php';  
        break;  
}
```

The controller (continued)

```
case 'send_mail':  
    // Get the data from the Mail View page  
    $from = $_POST['from'];  
    $to = $_POST['to'];  
    $subject = $_POST['subject'];  
  
    $text = $_POST['text'];  
    $url = $_POST['url'];  
    $message = $_POST['message'];  
  
    // Create the body  
    $body = $text . "\n\n" . $url . "\n\n" . $message;
```

The controller (continued)

```
try {  
    // Send the email  
    send_email($to, $from, $subject, $body);  
  
    // Display the Search view for the current query  
    $videos = search_youtube($query);  
    include 'search_view.php';  
} catch (Exception $e) {  
    $error = $e->getMessage();  
    include 'email_view.php';  
}  
break;
```

The controller (continued)

```
        default:
            include 'search_view.php';
            break;
    }

?>
```

Search view (search_view.php)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 ...>
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>YouTube Search</title>
    <link rel="stylesheet" type="text/css"
          href="main.css"/>
  </head>
  <body>
    <div id="page">
      <div id="header"><h1>YouTube Search</h1></div>
      <div id="main">
```

Search view (continued)

```
<h2>Search</h2>
<form action="." method="post">
    <input type="text" name="query"
        value="<?php echo $query; ?>" />
    <input type="submit" value="Search" />
</form>
<?php if (count($videos) != 0) : ?>
    <h2>Results</h2>
    <table>
        <?php foreach ($videos as $video) :
            $imgsrc =
                $video['media$group']['media$thumbnail'][0]['url'];
            $url = $video['link'][0]['href'];
            $text = $video['title']['$t'];
        ?>
        <tr>
            <td>
                <a href="<?php echo $url; ?>">
                    
                </a>
            </td>
```

Search view (continued)

```
<td>
    <form action="." method="post">
        <input type="hidden" name="action"
            value="display_email_view"/>
        <input type="hidden" name="url"
            value="<?php echo $url; ?>"/>
        <input type="hidden" name="text"
            value="<?php echo $text; ?>"/>
        <input type="submit"
            value="Email this Link"/>
    </form>
    <a href="<?php echo $url; ?>">
        <?php echo $text; ?>
    </a>
</td>
</tr>
<?php endforeach; ?>
</table>
<?php endif; ?>
```


Search view (continued)

```
        </div><!-- end main -->
    </div><!-- end page -->
</body>
</html>
```

Email view (mail_view.php)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 ...>
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>YouTube Search</title>
    <link rel="stylesheet" type="text/css"
          href="main.css"/>
  </head>
  <body>
    <div id="page">
      <div id="header"><h1>YouTube Search</h1></div>
      <div id="main">

<h2>Email the YouTube link</h2>
```

Email view (continued)

```
<?php if (isset($error)) : ?>
    <p>Error Sending E-mail: <?php echo $error; ?></p>
<?php endif; ?>

<form action="." method="post" id="email_form">
    <input type="hidden" name="action" value="send_mail"/>
    <input type="hidden" name="url"
        value="<?php echo $url; ?>" />
    <input type="hidden" name="text"
        value="<?php echo $text; ?>" />

    <label>From:</label>
    <input type="text" name="from"/>
    Your email address<br />
    <label>To:</label>
    <input type="text" name="to"/>
    Your friend's email address<br />
    <label>Subject:</label>
    <input type="text" name="subject"
        value="YouTube Video Link"/><br />
```

Email view (continued)

```
<label>Video:</label>  
<?php echo $text; ?><br />
```

```
<label>Link:</label>  
<?php echo $url; ?><br />
```

```
<label>Message:</label>  
<textarea name="message">  
    I thought you might enjoy this video.  
</textarea>  
<br />
```

```
<label>&nbsp;</label>  
<input type="submit" value="Send"/>  
</form>
```

Email view (continued)

```
<form action="." method="post" id="cancel_form">
    <label>&nbsp;</label>
    <input type="submit" value="Cancel"/>
</form>

    </div><!-- end main -->
</div><!-- end page -->
</body>
</html>
```