

CSCI 241

The Shopping Cart in an Online Store

*Department of CS & QM
Winthrop University
Fall 2011*

Prepared by X. Wang

Topics

- Online Store
- The Shopping Cart Management
- The Shopping Cart Implementation

Prepared by X. Wang

2

Online Store

- An online shopping store is a special online system that sells products over the Internet
- When a user browses an online store, the user may be interested in some products
- An online store should provide a tool for the user to pick up and keep the interested products
- This special tool is called the **shopping cart**

Prepared by X. Wang

3

The Shopping Cart Management

- The shopping cart is a necessary component in an online shopping store
- The user can **add items to the cart** and **manage the quantities** of different items
- The **shopping cart** must be **managed as a session** since the cart needs to be maintained for the user through many interactions once it is created
- Unlike the login session, the **cart data** needs to be **stored in database tables** during the shopping cart session

Prepared by X. Wang

4

The Shopping Cart Management

- To manage the shopping cart, we need to generate a **cart ID** for every shopping cart
- During the shopping cart session, we need to store this cart ID somewhere
- We can store the **cart ID** as **one session variable** in the **\$_SESSION** array
- We know that the shopping cart may exist **before or after** the customer logs in
- So the **shopping cart session** may exist in a **time range** that is **not the same as the login session**

Prepared by X. Wang

5

The Shopping Cart Management

- If the **cart ID** is stored as a session variable in **\$_SESSION**, we need to carefully handle the **\$_SESSION** array
- Particularly, we cannot simply **destroy the \$_SESSION** array when the customer **logs out**
- Instead, we can **only unset** the session variables that are related to the login session (for example, **cust_id**, **ip_addr**, **lastname** and **firstname** in our online bookstore)

Prepared by X. Wang

6

The Shopping Cart Management

- The second way, we can store the `cart ID` as a `separate cookie` at the client-side
- If the `cart ID` is stored as a cookie, we actually `separate the shopping cart session and login session` as two independent sessions
- In this chapter, we will discuss how to use a cookie to manage a session

Prepared by X. Wang

7

The Shopping Cart Management

- PHP provides the `setcookie()` function to allow us to set a cookie to the client-side (browser)


```
bool setcookie(string name [, string value [, int expire]])
```
- After setting a cookie, we can access the global array `$_COOKIE` with the `cookie name` as the associative key to get the `cookie value`
- For example, to set a shopping cart ID (`$cartid`) as a cookie with the name `cartid`, we can call


```
setcookie("cartid", $cartid);
```

Prepared by X. Wang

8

The Shopping Cart Management

- The `setcookie()` function must be called before the page sends out any output to the browser
- To get the value, we can do


```
$cartid = $_COOKIE["cartid"];
```
- If the `expire` time parameter is set, the cookie will `expire in the time`
- Otherwise, the cookie will `expire` when the `browser is closed` (a browser session is ended)

Prepared by X. Wang

9

The Cart Implementation

- To implement the shopping cart for an online store, we need to provide the following operations
 - Add items to the shopping cart
 - View the shopping cart
 - Update quantities of items on the shopping cart
 - Empty the shopping cart
 - Check out the shopping cart

Prepared by X. Wang

10

The Cart Implementation

- Add items to the shopping cart
- To add an item to the shopping cart, the script needs to handle three different cases
 - The shopping cart is empty, and then a new cart needs to be created before the item is added
 - A new item is added to the existing shopping cart
 - The item is already in the cart, and the script only adds an additional quantity to the item

Prepared by X. Wang

11

The Cart Implementation

- Add items to the shopping cart
- The `add-to-cart` page needs to accept an item ID (`book_id`) before the adding
- To start the `add-to-cart` page, we usually place an `active link` on or a `button` next to the item displayed in other pages like `Browse` and `Select`
- In the `add-to-cart` page, the script should fetch the item ID through the `$_GET` array since this page can be activated by an active link

Prepared by X. Wang

12

The Cart Implementation

- ❑ Add items to the shopping cart
- In our online bookstore, we have developed [Browse](#), [Search](#) and [Select](#) pages in Assignment-7 and 8
- In the [Select](#) page, we provide detailed information for a book item
- Also we need to provide an "Add to Cart" link or a button to allow the user to add the book to his shopping cart
- In order to show the process clearly, the PHP script for the [Select](#) page is listed here.

Prepared by X. Wang

13

The Cart Implementation

- The following [Select](#) page (select.php) can start an add-to-cart operation

```
<?php
require_once 'header.php';
require "db.php";

if (!($connection = @ mysql_connect($hostname, $dbusername, $dbpassword)))
{ print "Could not connect to database"; require_once 'footer.php'; exit; }

if (!(@ mysql_select_db($databasename, $connection))
{ showerror(); require_once 'footer.php'; exit; }

$isbn = $_GET["isbn"];
if (!isset($isbn) || empty($isbn))
{ print "<p><br /> Sorry, no book is found!</p>"; require_once 'footer.php'; exit; }

$query = "SELECT * FROM books WHERE ISBN={$isbn} OR ISBN13={$isbn}";
if (!($result = @ mysql_query ($query, $connection)))
{ showerror(); require_once 'footer.php'; exit; }
```

Prepared by X. Wang

14

The Cart Implementation

```
$num_result = mysql_num_rows($result);
if ($num_result == 0)
{ print "<p><br /> Sorry, no book is found!</p>"; require_once 'footer.php'; exit; }

$row = @ mysql_fetch_array($result);
print "<br /><img src='\"bookimages/{$row[\"ISBN\"]}.jpg\"' width='125' height='160'
align='left' />";

print "&nbsp; Title: <b>{$row[\"title\"]}</b><br />";
print "&nbsp; Author: {$row[\"authors\"]}<br />";
print "&nbsp; Publisher: {$row[\"publisher\"]}<br />";
print "&nbsp; Publish Date: {$row[\"publishdate\"]}<br />";
print "&nbsp; ISBN-10: {$row[\"ISBN\"]}<br />";
print "&nbsp; ISBN-13: {$row[\"ISBN13\"]}<br />";
print "&nbsp; Pages: {$row[\"pages\"]}<br />";
print "&nbsp; Price: <b>${$row[\"price\"]}</b><br />";
```

Prepared by X. Wang

15

The Cart Implementation

```
print "<form name='form1' id='form1' method='GET' action='addtocart.php'>";
$bookid = $row["book_id"];
$query2 = "SELECT sum(instock) FROM inventory WHERE book_id = {$bookid}";
if (!($result2 = @ mysql_query ($query2, $connection))) showerror();
else {
$row2 = @ mysql_fetch_array($result2);
$instock = $row2["sum(instock)"];
print "&nbsp; Status: ";
if ($instock > 0) {
print "<b>In Stock</b> &nbsp;";
print "<input type='hidden' name='bookid' value='\".$bookid.\"' />";
print "<input type='submit' value='Add to Cart' />";
}
else print "<b>Out of Stock</b> &nbsp;";
}
print "</forms>";
print "&nbsp; Description:<br /> {$row[\"description\"]}<br /><br />";
require_once 'footer.php';
?>
```

Prepared by X. Wang

16

The Cart Implementation

- The [Select](#) page presents the following output



Prepared by X. Wang

17

The Cart Implementation

- ❑ Add items to the shopping cart
- In the previous page, the user clicks the "Add to Cart" button, and the next PHP page ([addtocart.php](#)) will be started
- The Add-to-Cart page first checks whether there is an existing shopping cart by checking whether the `cartid` cookie has been set
- If no shopping cart, the script `inserts` a cart into the `carts` table in the database to create a new cart
- Then get the `cart ID` and `set the ID as a cookie`

Prepared by X. Wang

18

The Cart Implementation

- ❑ Add items to the shopping cart
- Otherwise, if there is already a cart, get the cart ID from the `$_COOKIE` array, and then check whether the book is already on the cart
- If not, insert a new item into the `cartitems` table and set the quantity to 1
- If the cart has the item, only increase the quantity by 1
- Note that how to get the `item_no` (primary key) from the `cartitems` table
- Finally, the page jump to `cartlist.php` to show the item list on the card (to avoid Reload problem)

Prepared by X. Wang

19

The Cart Implementation

- The following is the Add-to-Cart page (`addtocart.php`)

```
<?php ob_start();
require_once 'header.php';
require 'db.php';
if (!$connection = @ mysql_connect($hostname, $dbusername, $dbpassword))
{ print "Could not connect to database"; require_once 'footer.php'; exit; }
if (!(@ mysql_select_db($databasename, $connection))
{ showerror(); require_once 'footer.php'; exit; }
if (!isset($_COOKIE['cartid'])) {
    $query = "INSERT INTO carts SET datecreated = now()";
    if (!($result = @ mysql_query ($query, $connection)))
    { showerror(); require_once 'footer.php'; exit; }
    $cartid = mysql_insert_id();
    setcookie("cartid", $cartid);
} else $cartid = $_COOKIE['cartid'];
$bookid = $_GET['bookid'];
$query = "SELECT * FROM cartitems WHERE cart_id={$cartid} AND book_id={$bookid}";
if (!($result = @ mysql_query ($query, $connection))
{ showerror(); require_once 'footer.php'; exit; }
```

Prepared by X. Wang

20

The Cart Implementation

```
$num_result = mysql_num_rows($result);
if ($num_result == 0) { // the book is not on the cart
    $query = "SELECT MAX(item_no) FROM cartitems WHERE cart_id={$cartid}";
    if (!($result = @ mysql_query ($query, $connection))
    { showerror(); require_once 'footer.php'; exit; }
    $row = @ mysql_fetch_array($result);
    $item_no = $row['MAX(item_no)']+1;
    $query = "INSERT INTO cartitems VALUES ($cartid, $item_no, 1, $bookid, now())";
    if (!($result = @ mysql_query ($query, $connection))
    { showerror(); require_once 'footer.php'; exit; }
} else { // the book is on the cart already, and only increase quantity
    $query = "UPDATE cartitems SET quantity = quantity + 1 WHERE cart_id={$cartid}
AND book_id={$bookid}";
    if (!($result = @ mysql_query ($query, $connection))
    { showerror(); require_once 'footer.php'; exit; }
}
header("LOCATION: cartlist.php");
?>
```

Prepared by X. Wang

21

The Cart Implementation

- ❑ View the shopping cart
- After an item is added to the shopping cart, the script should automatically display all items on the cart
- We usually use an HTML table to list all items on the cart
- The quantity of each item allows the user to change by giving a new number and clicking the Update button
- The shopping cart view page can be accessed explicitly by typing the URL address or clicking an active link
- For example, in our online bookstore, clicking the "Shopping Cart" link in the navigation bar will directly access this page

Prepared by X. Wang

22

The Cart Implementation

- The following is the `cartlist.php` page

```
<?php
require_once 'header.php';
require 'db.php';

if (!isset($_COOKIE['cartid']))
{ print "<br /><b>The Shopping Cart is Empty!</b>"; require_once 'footer.php'; exit; }

$cartid = $_COOKIE['cartid'];
if (!$connection = @ mysql_connect($hostname, $dbusername, $dbpassword))
{ print "Could not connect"; require_once 'footer.php'; exit; }
if (!(@ mysql_select_db($databasename, $connection))
{ showerror(); require_once 'footer.php'; exit; }

$query = "SELECT * FROM cartitems, books
WHERE cart_id={$cartid} AND cartitems.book_id=books.book_id
ORDER BY item_no";
if (!($result = @ mysql_query ($query, $connection))
{ showerror(); require_once 'footer.php'; exit; }
```

Prepared by X. Wang

23

The Cart Implementation

```
$num_result = mysql_num_rows($result);
if ($num_result == 0)
{ print "<br /><b>The Shopping Cart is Empty!</b>"; require_once 'footer.php'; exit; }

print '<table border="0" width="100%">';
print '<tr><td height="16"><b>Shopping Cart</b></td>';
print '<td align="right"><form name="form1" id="form1" method="post"
action="checkout.php">';
print '<input type="submit" value="Check Out" />';
print '</forms></td></tr>';
print '</table>';

print '<table bgcolor="#CCDDFF" width="100%" border="1">';
print '<form name="form2" id="form2" method="post" action="updatecart.php">';
print '<tr><td width="90">Date Added:</td><td>Item Description</td>';
print '<td>Unit Price</td><td>Qty</td><td align="right" width="40">Total</td></tr>';
print '<td colspan="4" style="text-align: right; padding-top: 5px;">Total = 0.0;
```

Prepared by X. Wang

24

The Cart Implementation

```

for ($i=1; $i<=$num_result; $i++) {
    $row = @ mysql_fetch_array($result);
    print "<tr><td>{$row['dateadded']}</td>";
    print "<td><a href='select.php?isbn={$row['ISBN']}'>{$row['title']}</a><br />";
    print " - {$row['authors']}<br />ISBN-10: {$row['ISBN']}<br />ISBN-13:
    {$row['ISBN13']}</td>";
    print "<td align='right'>{\$row['price']}</td>";
    print "<td><input type='text' name='{$row['item_no']}' value='{$row['quantity']}'
    size='1' /></td>";
    $subtotal = $row['price'] * $row['quantity'];
    print "<td align='right'>{\$subtotal}</td>";
    print "</tr>";
    $total += $subtotal;
}
print "<tr><td colspan='2'>&nbsp;&nbsp;&nbsp;</td>";
print "<td colspan='2' align='right'><input type='submit' value='Update' /></td>";
print "<td align='right'>{\$total}</td>";
Print "</tr></form></table>";
require_once 'footer.php';
?>

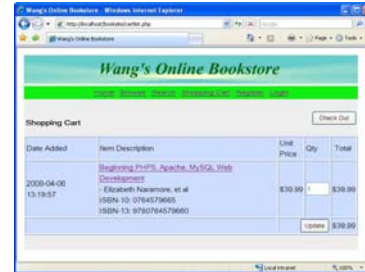
```

Prepared by X. Wang

25

The Cart Implementation

- Here is the shopping cart list page



Prepared by X. Wang

26

The Cart Implementation

- ❑ View the shopping cart
- In the previous PHP script, we used a text field to display the item quantities
- Each field is named with the item-no (primary key)
- This name will allow the Update page to easily find the corresponding item on the cart and update the quantity for the item in the table
- In this view page, we also provide an HTML form button "Check Out" to allow the customer to go to the checkout page

Prepared by X. Wang

27

The Cart Implementation

- ❑ Update item quantities on the shopping cart
- If the user changes the item quantities on the cart list and clicks the Update button, all quantities will be sent to the updatecart.php page with the POST form method
- If the new quantity is 0, the item should be removed from the cartitems table
- If no item is on the cart, the cart should be removed from the carts table in the database
- If the new quantity is larger than the number of books in stock, the script should reject the update

Prepared by X. Wang

28

The Cart Implementation

- The following is the page updatecart.php

```

<?php ob_start();
require_once 'header.php';
require 'db.php';

if (!isset($_COOKIE['cartid']))
{ print "Invalid operation!!!"; require_once 'footer.php'; exit; }
$cartid = $_COOKIE['cartid'];
if (!($connection = @ mysql_connect($hostname, $dbusername, $dbpassword)))
{ print "Could not connect to database"; require_once 'footer.php'; exit; }

if (!(@ mysql_select_db($databasename, $connection)))
{ showerror(); require_once 'footer.php'; exit; }

$query = "SELECT item_no, sum(instock) FROM cartitems c, inventory v
WHERE cart_id={$cartid} AND c.book_id = v.book_id
GROUP BY item_no";
if (!($result = @ mysql_query ($query, $connection)))
{ showerror(); require_once 'footer.php'; exit; }

```

Prepared by X. Wang

29

The Cart Implementation

```

$num_result = mysql_num_rows($result);
for ($i=1; $i<=$num_result; $i++) {
    $row = @ mysql_fetch_array($result);
    $item_no = $row['item_no'];
    if (isset($_POST['{$item_no}'])) {
        $quantity = (int) $_POST['{$item_no}'];
        if ($quantity <= $row['sum(instock)']) {
            if ($quantity > 0)
                $query2 = "UPDATE cartitems SET quantity={$quantity}";
            else
                $query2 = "DELETE FROM cartitems";
            $query2 .= " WHERE cart_id={$cartid} AND cartitems.item_no={$item_no}";
            if (!($result2 = @ mysql_query ($query2, $connection)))
                { showerror(); require_once 'footer.php'; exit; }
        }
    }
}

```

Prepared by X. Wang

30

The Cart Implementation

```
$query = "SELECT * FROM cartitems WHERE cart_id={$cartid}";
if (!($result = @ mysql_query ($query, $connection)))
{ showerror(); require_once 'footer.php'; exit; }

$num_result = mysql_num_rows($result);
if ($num_result == 0) {
    $query = "DELETE FROM carts WHERE cart_id={$cartid}";
    if (!($result = @ mysql_query ($query, $connection)))
    { showerror(); require_once 'footer.php'; exit; }
    setcookie("cartid", $cartid, time()-3600);
}
header("LOCATION: cartlist.php");
?>
```

Prepared by X. Wang

31

The Cart Implementation

- ❑ Check out the shopping cart
- After adding all items to the shopping cart, the user may want to make the purchase
- So on the cart list page, we provide a "Check Out" button to allow the user to check out
- The Check-Out operation only works for a customer who has logged in the system
- The script checks whether the user is already in the login session by checking the `$_SESSION["cust_id"]` session variable

Prepared by X. Wang

32

The Cart Implementation

- ❑ Check out the shopping cart
- If a user has not logged in, clicking the **CheckOut** button will be directed to the login page
- In the **CheckOut** page (`checkout.php`), the script displays an HTML form to let the user enter the **credit card** information
- The page also lists all the items in the current cart so that the user can double-check his cart information

Prepared by X. Wang

33

The Cart Implementation

- Let's see the **CheckOut** page (`checkout.php`)

```
<?php
require_once 'header.php';
require 'db.php';

if (!isset($_COOKIE["cartid"]))
{ print "<br /><b>The Shopping Cart is Empty!</b>"; require_once 'footer.php'; exit; }
$cartid = $_COOKIE["cartid"];

if (!($connection = @ mysql_connect($hostname, $dbusername, $dbpassword)))
{ print "Could not connect to database"; require_once 'footer.php'; exit; }
if (!(@ mysql_select_db($database, $connection)))
{ showerror(); require_once 'footer.php'; exit; }

$query = "SELECT * FROM cartitems, books
WHERE cart_id={$cartid} AND cartitems.book_id=books.book_id
ORDER BY item_no";
if (!($result = @ mysql_query ($query, $connection)))
{ showerror(); require_once 'footer.php'; exit; }
```

Prepared by X. Wang

34

The Cart Implementation

```
$num_result = mysql_num_rows($result);
if ($num_result == 0)
{ print "<br /><b>Shopping Cart is Empty!</b>"; require_once 'footer.php'; exit; }

if (!isset($_SESSION["cust_id"]))
{ print "<br /><b>To check out, you must <a href='login.php'>login</a> your account first</b>"; require_once 'footer.php'; exit; }

print "<br /><b>Please provide your Credit Card information:</b><br />";
print "<table border='0' width='100%'>";
print "<tr><td colspan='2'><input type='text' name='creditcard' size='22' /></td></tr>";
print "<tr><td align='right'>Expiry Date(<input type='text' name='expirymonth' size='11' /> /<input type='text' name='expiryyear' size='11' /> (mm/yy)</td></tr>";
print "<tr><td align='right'>&nbsp;&nbsp;&nbsp;<input type='submit' value='Proceed the Order' /></td></tr>";
print "</form></table>";
```

Prepared by X. Wang

35

The Cart Implementation

```
print "<br /><b>Current Shopping Cart</b><br />";
print "<table class='backcolor2' width='100%' border='1'>";
print "<tr><td width='90%'>Date Added</td><td>Item Description</td>";
print "<td width='50%'>Unit Price</td><td>Qty</td><td align='right' width='50%'>Total</td></tr>";
$total = 0.0;
for ($i=1; $i<=$num_result; $i++) {
    $row = @ mysql_fetch_array($result);
    print "<tr><td>{$row['dateadded']}</td>";
    print "<td><a href='select.php?isbn={$row['ISBN']}'>{$row['title']}</a><br />";
    print "<td>{$row['authors']}<br />ISBN-10: {$row['ISBN']}<br />ISBN-13: {$row['ISBN13']}</td>";
    print "<td align='right'>{$row['price']}</td>";
    print "<td>{$row['quantity']}</td>";
    $subtotal = $row['price'] * $row['quantity'];
    print "<td align='right'>{$subtotal}</td>";
    print "</tr>";
    $total += $subtotal;
}
print "<br />Total: $total";
```

Prepared by X. Wang

36

The Cart Implementation

```
print '<tr><td colspan="2">&nbsp;</td>';
print '<td colspan="2" align="right">&nbsp;</td>';
print '<td align="right">${$Total}</td>';
print '</tr>';
print '</table>';
```

```
require_once 'footer.php';
?>
```

Prepared by X. Wang

37

The Cart Implementation

- The checkout.php page displays the following interface



Prepared by X. Wang

38

The Cart Implementation

- ❑ Check out the shopping cart
- After entering the credit card information and clicking the "Proceed the Order" button, the user entered information will be sent to the page ([makeorder.php](#)) to do credit card validation
- The [makeorder.php](#) page will validate the credit card number by calling the `checkCreditCard()` function introduced in the topic "Validating User Input"
- This page can only be accessed by the logged-in user

Prepared by X. Wang

39

The Cart Implementation

- ❑ Check out the shopping cart
- If passing the validation, the script inserts an order in the `orders` table, and copies all items on the current shopping cart in the `cartitems` table to the `items` table
- Then the `shopping cart` is set to empty by deleting all items from the `cartitems` table and `carts` table
- Finally, the page jumps to the [showorders.php](#) page to list all orders made by the user
- This also can avoid the Reload problem

Prepared by X. Wang

40

The Cart Implementation

- The following is the [makeorder.php](#) page

```
<?php
ob_start();
require_once 'header.php';
require 'db.php';

if (!isset($_SESSION['cust_id']) || !isset($_COOKIE['cartid']))
{ print "Invalid operation!!!"; require_once 'footer.php'; exit; }
$cust_id = $_SESSION['cust_id'];
$cartid = $_COOKIE['cartid'];

if (!($connection = @ mysql_connect($hostname, $dbusername, $dbpassword)))
{ print "Could not connect to database"; require_once 'footer.php'; exit; }

if (!(@ mysql_select_db($databasename, $connection)))
{ showerror(); require_once 'footer.php'; exit; }

$query = "SELECT * FROM cartitems WHERE cart_id={$cartid} ORDER BY item_no";
if (!($result = @ mysql_query ($query, $connection)))
{ showerror(); require_once 'footer.php'; exit; }
```

Prepared by X. Wang

41

The Cart Implementation

```
$num_result = mysql_num_rows($result);
if ($num_result == 0)
{ print "Invalid operation!!!"; require_once 'footer.php'; exit; }

$creditcard = $_POST['creditcard'];
$retocode = checkCreditCard($creditcard);
if ($retocode != 0) {
switch ($retocode) {
case 1: print "Credit Card field is blank"; break;
case 2: print "Card number must contain only digits and spaces"; break;
case 3: print "Invalid card number<br />->Please check the number"; break;
}
require_once 'footer.php'; exit;
}
if (empty($_POST['expirymonth']))
{ print "Expiry Month is blank"; require_once 'footer.php'; exit; }
if (empty($_POST['expiryyear']))
{ print "Expiry Year is blank"; require_once 'footer.php'; exit; }
$expirydate = $_POST['expirymonth'] . $_POST['expiryyear'];
```

Prepared by X. Wang

42

The Cart Implementation

- Finally, the user sees the following order list

The screenshot shows a web browser window titled "Wang's Online Bookstore". The page displays two orders. The first order, "Order #1 Made on 2008-04-10 10:20:00 Processing", lists two items: "Stephen G. Crane's Red Badge This Side of Paradise" (2 units at \$450.00 each) and "Stephen Crane's Red Badge This Side of Paradise" (1 unit at \$20.00). The second order, "Order #2 Made on 2008-04-09 13:42:02 Processing", lists one item: "Stephen Crane's Red Badge This Side of Paradise" (1 unit at \$20.00). Each item row includes columns for "Name", "Description", "Unit Price", "Qty", and "Total".

Order #	Made on	Status	Name	Description	Unit Price	Qty	Total
Order #1	2008-04-10 10:20:00	Processing	Stephen G. Crane's Red Badge This Side of Paradise	Stephen G. Crane's Red Badge This Side of Paradise	\$450.00	2	\$900.00
Order #1	2008-04-10 10:20:00	Processing	Stephen Crane's Red Badge This Side of Paradise	Stephen Crane's Red Badge This Side of Paradise	\$20.00	1	\$20.00
Order #2	2008-04-09 13:42:02	Processing	Stephen Crane's Red Badge This Side of Paradise	Stephen Crane's Red Badge This Side of Paradise	\$20.00	1	\$20.00

Prepared by X. Wang