

Instalación Smarty

Para instalar Smarty en el equipo es necesario tener previamente instalado php y un servidor local.

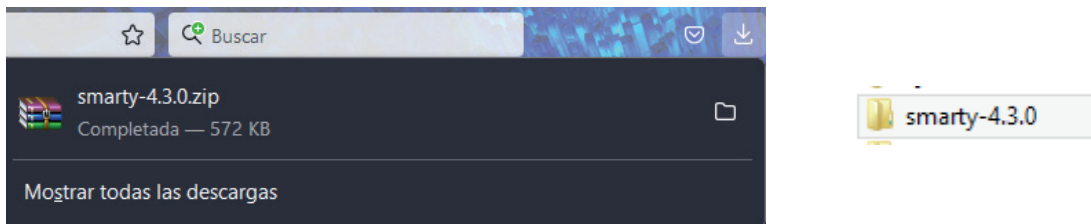
Primero descargamos el programa desde su página web, <https://www.smarty.net/>. Elegimos una versión por debajo de la actual, ya que esta suele ser más estable que la más reciente. En mi caso elegí la versión 4.3.0.

The screenshot shows the Smarty Template Engine website. The main content area is titled "Smarty 4.3.2 released!" and "Smarty 4.3.0 and 3.1.48 Released!". A red box highlights the "4.3.0" link and the "v4_Change Log" link. The right sidebar contains a "Get Smarty" section with links to "Download", "About Smarty", "Why use it?", "Use Cases and Work Flow", "Syntax Comparison", "Template Inheritance", "Best Practices", "Crash Course", "Version 3 Overview", "Testimonials", and "Resources".

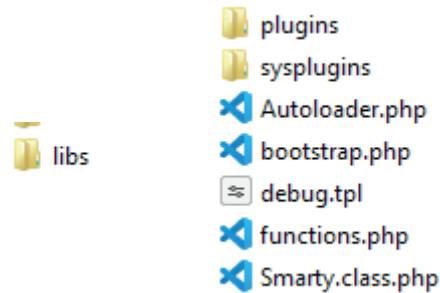
Hacemos click sobre el enlace y este no llevará a un repositorio de GitHub. Si bajamos hasta el final de la página encontraremos el .zip que debemos descargar.

The screenshot shows the GitHub release page for Smarty v4.3.0. The release was made by wisskid on Nov 22, 2022. The "What's Changed" section lists several updates, including clean output buffer for Throwable, fix wrong indentation in libs/plugins/modifier.capitalize.php, and fix compilation for caching templates. The "New Contributors" section lists several contributors, including MrPetovan, Storyxx, asmecher, EDCScott, MekDrop, AndrewDawes, and Progi1984. The "Assets" section shows two files: "Source code (zip)" and "Source code (tar.gz)".

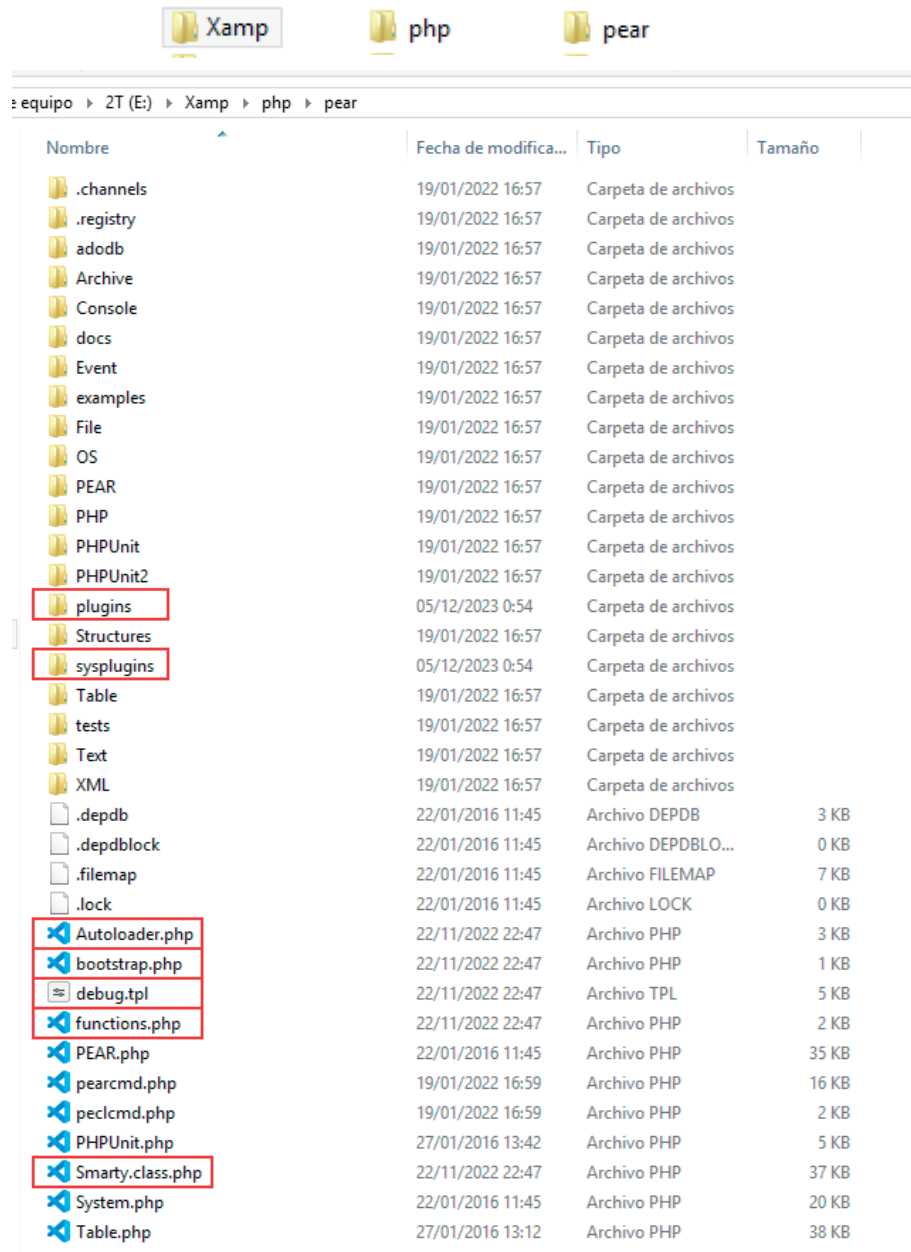
Lo descargamos y extraemos el contenido del .zip en una carpeta a nuestro gusto.



Ahora, dentro de la carpeta Smarty, abrimos la carpeta libs, seleccionamos todo su contenido y lo copiamos.



Lo siguiente es ir a Xampp → php → pear y pegar todo el contenido que habíamos copiado.



Comprobar la instalación

Activamos Xampp para poder ejecutar php en el navegador y poder comprobar que la instalación se ha hecho adecuadamente.

Lo siguiente que debemos hacer para completar la instalación es cambiar las rutas de los archivos .php de la plantilla con la que haremos el ejercicio.

Estado original de las rutas

```
require_once('Smarty.class.php');
$smarty = new Smarty;
$smarty->template_dir = '/web/smarty/tarea/templates/';
$smarty->compile_dir = '/web/smarty/tarea/templates_c/';
$smarty->config_dir = '/web/smarty/tarea/configs/';
$smarty->cache_dir = '/web/smarty/tarea/cache/';
```

Estado odificado de las rutas

```
require_once('Smarty.class.php');
$smarty = new Smarty;
$smarty->template_dir = '../web/smarty/tarea/templates/';
$smarty->compile_dir = '../web/smarty/tarea/templates_c/';
$smarty->config_dir = '../web/smarty/tarea/configs/';
$smarty->cache_dir = '../web/smarty/tarea/cache/';
```

En el navegador buscamos el localhost y accedemos a la plantilla del ejercicio que se nos ha dado. Abrimos el archivo login.

Login

Warning: Undefined array key "error" in E:\Xamp\htdocs\PHP\TAREAS\DWES05\ejercicioecuelto\web\smarty\stienda\templates_c\395d9f9a0c68d9bb8cc01e2f53f512f780c413dd_0.file.login.tpl.php on line 40

Warning: Attempt to read property "value" on null in E:\Xamp\htdocs\PHP\TAREAS\DWES05\ejercicioecuelto\web\smarty\stienda\templates_c\395d9f9a0c68d9bb8cc01e2f53f512f780c413dd_0.file.login.tpl.php on line 40

Usuario:

Contraseña:

Enviar

Nos sale un error, pero podemos ver que la plantilla funciona y se visualiza.

Vamos hasta donde el error nos indica, pero el archivo al que nos dirige el error es un tipo de archivo que crea Smarty, después de crear nosotros un template junto con su php, por lo que el error debe ser en alguno de estos dos archivos.

```
<legend>Login</legend>
<div><span class='error'><?php echo $_smarty_tpl->tpl_vars['error']->value;?></span></div>
<div class='campo'>
```

Después de mirar en login.tpl y poder ver que las variables que usa esta plantilla provienen del php, vamos a login.php. Aquí podemos ver la variable error que está dando problemas.

```
<?php
require_once('include/DB.php');

// Cargamos la librería de Smarty
require_once('Smarty.class.php');
$smarty = new Smarty;
$smarty->template_dir = '../web/smarty/stienda/templates/';
$smarty->compile_dir = '../web/smarty/stienda/templates_c/';
$smarty->config_dir = '../web/smarty/stienda/configs/';
$smarty->cache_dir = '../web/smarty/stienda/cache/';

// Comprobamos si ya se ha enviado el formulario
if (isset($_POST['enviar'])) {

    if (empty($_POST['usuario']) || empty($_POST['password']))
        $smarty->assign('error','Debes introducir un nombre de usuario y una contraseña');
    else {
        // Comprobamos las credenciales con la base de datos
        if (DB::verificaCliente($_POST['usuario'], $_POST['password'])) {
            session_start();
            $_SESSION['usuario']=$_POST['usuario'];
            header("Location: productos.php");
        }
        else {
            // Si las credenciales no son válidas, se vuelven a pedir
            $smarty->assign('error','Usuario o contraseña no válidos!');
        }
    }
}

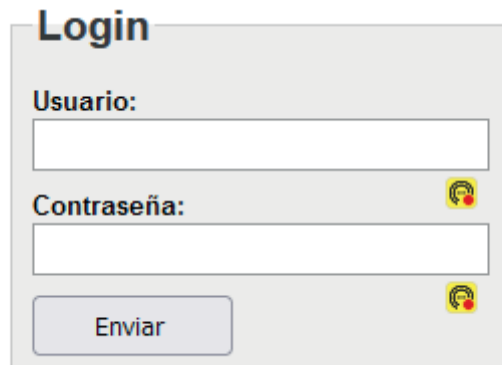
// Mostramos la plantilla
$smarty->display('login.tpl');
?>
```

Simplemente le damos un valor vacío para que deje de salir el error. Quedando así el código.

```
<?php
require_once('include/DB.php');
// Cargamos la librería de Smarty
require_once('Smarty.class.php');
$smarty = new Smarty;
$smarty->template_dir = '../web/smarty/tarea/templates/';
$smarty->compile_dir = '../web/smarty/tarea/templates_c/';
$smarty->config_dir = '../web/smarty/tarea/configs/';
$smarty->cache_dir = '../web/smarty/tarea/cache/';
/*--Modificación frente al original--*/
//creación de la variable error
$smarty->assign('error','');
$error='';
/*--FIN Modificación frente al original--*/
if (isset($_POST['enviar'])) {
    if (empty($_POST['usuario']) || empty($_POST['password']))
        $smarty->assign('error','Debes introducir un nombre de usuario y una contraseña');
    else {
        // Comprobamos las credenciales con la base de datos
        if (DB::verificaCliente($_POST['usuario'], $_POST['password'])) {
            session_start();
            $_SESSION['usuario']=$_POST['usuario'];
            header("Location: productos.php");
        }
        else {
            // Si las credenciales no son válidas, se vuelven a pedir
            $smarty->assign('error','Usuario o contraseña no válidos!');
        }
    }
}

// Mostramos la plantilla
$smarty->display('login.tpl');
?>
```

Ahora volvemos al navegador y recargamos para ver si los cambios realizados han surtido el efecto deseado. Vemos que así es. Con esto ya, podemos empezar a realizar el ejercicio.



The image shows a web form titled "Login". It has two input fields: "Usuario:" and "Contraseña:". Below the "Contraseña:" field is a button labeled "Enviar". There are small circular icons to the right of the password field and below the "Enviar" button.

Creación de la Base de Datos. DB_Create.php

Para crear la base de datos primero debemos conectarnos con el servidor local.

```
//Creamos la conexión con el servidor con el usuario root para poder crear la base de datos
$enlace = new PDO('mysql:host=localhost','root','root');
```

Al usar este archivo (DB_Create.php) quizás sea necesario cambiar la contraseña del usuario root en el código para que pueda funcionar.

Una vez que sepamos que esta conexión se ha establecido, borramos la base de datos que haya con el mismo nombre que la que vamos a utilizar, ya que no sabemos si esta ha sido modificada de alguna manera que pueda interferir con el desarrollo del ejercicio. Además en este caso no habría problema de pérdida de información, pues esta base de datos no será modificada durante el ejercicio.

```
//borramos la base de datos para volver a crearla, en este ejercicio es posible hacerlo
//ya que la base de datos no cambia su información durante el desarrollo del ejercicio
$enlace->exec("DROP DATABASE IF EXISTS `tarea5`");
```

Lo siguiente es crear la base de datos y el usuario. También le daremos los privilegios necesarios a este para poder interactuar con la información sin problemas.

```
//sentencia SQL para crear la bdd, También creamos el usuario y le damos toso los privilegios
$enlace->exec("CREATE DATABASE IF NOT EXISTS `tarea5` DEFAULT CHARACTER SET utf8 COLLATE utf8_spanish_ci;
CREATE USER IF NOT EXISTS `dwes` IDENTIFIED BY 'abc123.';
GRANT ALL ON `tarea5`.* TO 'dwes';
FLUSH PRIVILEGES;");
```

Ahora es el momento de conectarnos a la base de datos con el usuario y contraseña que hemos creado antes y ver si lo hemos hecho correctamente.

```
//nos conectamos a la base de datos creada anteriormente
$db = new PDO('mysql:host=localhost;dbname=tarea5','dwes','abc123.');
```

Como no nos ha salido ningún error procedemos a crear la primera tabla – Familia –

```
/*----- TABLA FAMILIA -----*/
//sentencia SQL para crear la tabla FAMILIA
$sql= "CREATE TABLE IF NOT EXISTS `familia` (
`cod` varchar(6) COLLATE utf8_spanish_ci NOT NULL,
`nombre` varchar(200) COLLATE utf8_spanish_ci NOT NULL,
PRIMARY KEY (`cod`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_spanish_ci;";
//ejecutamos la sentencia
$db->exec($sql);
```

Después la rellenamos con la información necesaria.

```
//Volcar la base de datos para la tabla `familia`
$sql="INSERT INTO `familia` (`cod`, `nombre`) VALUES
('CAMARA', 'Cámaras digitales'),
('CONSOL', 'Consolas'),
('EBOOK', 'Libros electrónicos'),
('IMPRES', 'Impresoras'),
('MEMFLA', 'Memorias flash'),
('MP3', 'Reproductores MP3'),
('MULTIF', 'Equipos multifunción'),
('NETBOK', 'Netbooks'),
('ORDENA', 'Ordenadores'),
('PORTAT', 'Ordenadores portátiles'),
('ROUTER', 'Routers'),
('SAI', 'Sistemas de alimentación ininterrumpida'),
('SOFTWA', 'Software'),
('TV', 'Televisores'),
('VIDEOC', 'Videocámaras');";
//ejecutamos la sentencia sql
$db->exec($sql);
/*----- FIN TABLA FAMILIA -----*/
```

Realizamos este mismo proceso con la tabla – Producto –.

```
/*----- TABLA PRODUCTO -----*/
//sentencia SQL para crear la tabla PRODUCTO
$sql= "CREATE TABLE IF NOT EXISTS `producto` (
`cod` varchar(12) COLLATE utf8_spanish_ci NOT NULL,
`nombre` varchar(200) COLLATE utf8_spanish_ci DEFAULT NULL,
`nombre_corto` varchar(50) COLLATE utf8_spanish_ci NOT NULL,
`descripcion` text COLLATE utf8_spanish_ci,
`PVP` decimal(10,2) NOT NULL,
`familia` varchar(6) COLLATE utf8_spanish_ci NOT NULL,
PRIMARY KEY (`cod`),
UNIQUE KEY `nombre_corto` (`nombre_corto`),
KEY `familia` (`familia`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_spanish_ci;";
//ejecutamos la sentencia
$db->exec($sql);
```


Después la rellenamos con la información necesaria. Este fragmento de código no se ve entero pues es muy largo.

```

--Volcar la base de datos para la tabla 'producto'
$sql='INSERT INTO `producto` (`cod`,`nombre`,`nombre_corto`,`descripcion`,`PVP`,`familia`) VALUES
('3DSNG', NULL, 'Nintendo 3DS negro', 'Consola portátil de Nintendo que permitirá disfrutar de efectos 3D sin necesidad de gafas especiales, e incluirá retrocompatibilidad
con el software de DS y de DSi.', '270.00', 'CONSOL'),
('ACERAX3950', NULL, 'Acer AX3950 15-650 4GB 1TB W7HP', 'Características:\r\n\r\nSistema Operativo : Windows® 7 Home Premium Original\r\n\r\nProcesador / Chipset\r\nNúmero
de Ranuras PCI: 1\r\n\r\nFabricante de Procesador: Intel\r\n\r\nTipo de Procesador: Core i5\r\n\r\nModelo de Procesador: i5-650\r\n\r\nNúcleo de Procesador: Dual-core\r\n\r\nVelocidad de
Procesador: 3,20 GHz\r\n\r\nCache: 4 MB\r\n\r\nVelocidad de Bus: No aplicable\r\n\r\nVelocidad HyperTransport: No aplicable\r\n\r\nInterconexión QuickPath: No aplicable\r\n\r\nProcesamiento de
64 bits: Si\r\n\r\nHyper-Threading: Si\r\n\r\nFabricante de Chipset: Intel\r\n\r\nModelo de Chipset: H57 Express\r\n\r\n\r\nMemoria\r\nMemoria Estándar: 4 GB\r\n\r\nMemoria Máxima: 8
GB\r\n\r\nTecnología de la Memoria: DDR3 SDRAM\r\n\r\nEstándar de Memoria: DDR3-1333/PC3-10600\r\n\r\nNúmero de Ranuras de Memoria (Total): 4\r\n\r\nSelector de tarjeta memoria:
Si\r\n\r\nSoporte de Tarjeta de Memoria: Tarjeta CompactFlash (CF)\r\n\r\n\r\nSoporte de Tarjeta de Memoria: MultiMediaCard (MMC)\r\n\r\n\r\nSoporte de Tarjeta de Memoria: Micro
Drive\r\n\r\n\r\nSoporte de Tarjeta de Memoria: Memory Stick PRO\r\n\r\n\r\nSoporte de Tarjeta de Memoria: Memory Stick\r\n\r\n\r\nSoporte de Tarjeta de Memoria: CF+\r\n\r\n\r\nSoporte de Tarjeta de
Memoria: Tarjeta Secure Digital (SD)\r\n\r\n\r\n\r\nStorage\r\n\r\nCapacidad Total del Disco Duro: 1 TB\r\n\r\nRPM de Disco Duro: 5400\r\n\r\n\r\nTipo de Unidad Óptica: Grabadora
DVD\r\n\r\nCompatibilidad de Dispositivo Óptico: DVD-RAM/±RW\r\n\r\n\r\nCompatibilidad de Medios de Doble Capa: Si, '410.00', 'ORDENA'),
('ARCLPMP32GBN', NULL, 'Archos Clipper MP3 2GB negro', 'Características:\r\n\r\n\r\nAlmacenamiento Interno Disponible en 2 GB*\r\n\r\nCompatibilidad Windows o Mac y Linux (con
soporte para almacenamiento masivo)\r\n\r\n\r\nInterfaz para ordenador USB 2.0 de alta velocidad\r\n\r\nBatería21 horas música\r\n\r\nReproducción Música3 MP3\r\n\r\nMedidas Dimensiones:
52mm x 27mm x 12mm, Peso: 14 Gr., '26.70', 'MP3'),
('BRAVIA2BX400', NULL, 'Sony Bravia 32IN FULLHD KDL-32BX400', 'Características:\r\n\r\n\r\nFull HD: Vea deportes películas y juegos con magníficos detalles en alta resolución
gracias a la resolución 1920x1080.\r\n\r\n\r\nEntradas 3 en la parte posterior, 1 en el lateral\r\n\r\n\r\nUSB Media Player: Disfrute de películas, fotos y música en el
televisor.\r\n\r\n\r\nSintonizador de TV HD MPEG-4 AVC integrado: olvídense del codificador y acceda a servicios de TV que incluyen canales HD con el sintonizador DVB-T y DVB-C
integrado con decodificador MPEG4 AVC (dependiendo del país y sólo con operadores compatibles)\r\n\r\n\r\nSensor de luz: ajusta automáticamente el brillo según el nivel de la
iluminación ambiental para que pueda disfrutar de una calidad de imagen óptima sin consumo innecesario de energía.\r\n\r\n\r\n\r\nBRAVIA Sync: controle su sistema de ocio doméstico
entero con un mismo mando a distancia universal que le permite reproducir contenidos o ajustar la configuración de los dispositivos compatibles con un solo botón.
\r\n\r\n\r\nBRAVIA ENGINE 2: experimente colores y detalles de imagen increíblemente nítidos y definidos. \r\n\r\n\r\n\r\nLive Colour™: seleccione entre cuatro modos: desactivado,
bajo, medio y alto, para ajustar el color y obtener imágenes vivas y una calidad óptima. \r\n\r\n\r\n\r\n24p True Cinema™: reproduzca una auténtica experiencia cinematográfica y

```

```
((C:\SSMSMX\C200PB', NULL, 'Samsung SPM-C200PB ED ZOOM 10X', 'Características:\r\n\r\nSensor de imagen tipo 1 / 6" 880K pixel CCD\r\n\r\nLente zoom Óptico 10 x  
óptico\r\n\r\nCaracterísticas Grabación Vídeo Estabilizador de Imagen Hiper estabilizador de imagen digital\r\n\r\n\r\nInterfaz Tarjeta de Memoria Ranura de Tarjeta SDHC / SD',  
'127.20', 'VIDEOC'),  
(('STYLUSSX515W', NULL, 'Epson Stylus SX515W', 'Características:\r\n\r\nResolución máxima5760 x 1440 DPI\r\n\r\nVelocidad de la impresión\r\n\r\nVelocidad de impresión (negro,  
calidad normal, A4)36 ppm\r\n\r\nVelocidad de impresión (color, calidad normal, A4)36 ppm\r\n\r\nTecnología de la impresión\r\n\r\nTecnología de impresión inyección de  
tinta\r\n\r\nNúmero de cartuchos de impresión4 piezas\r\n\r\nCabeza de impresoraMicro Piezo\r\n\r\n\r\nExploración\r\n\r\nResolución máxima de escaneado2400 x 2400 DPI\r\n\r\nEscaner color:  
si\r\n\r\nTipo de digitalización Escáner plano\r\n\r\nEscáncera integrado: si\r\n\r\nTecnología de exploración CIS\r\n\r\n\r\nWLAN, conexión: si', '77.50', 'MULTITIF'),  
(('TSDS16GBIC107', NULL, 'Toshiba SD16GB Class10 Jewel Case', 'Características:\r\n\r\n\r\nDensidad: 16 GB\r\n\r\n\r\nPINPS de conexión: 9 pins\r\n\r\n\r\nInterfaz: Tarjeta de memoria SD standard  
compatible\r\n\r\nVelocidad de Escritura: 20 MBbytes/s* \r\n\r\nVelocidad de Lectura: 20 MBbytes/s*\r\n\r\nDimensiones: 32.0 mm (L) [ ] 24.0 mm (W) [ ] 2.1 mm (H)\r\n\r\nPeso: 2g\r\n\r\nTemperatura:  
-25°C a +85°C (Recomendada)\r\n\r\nHumedad: 30% to 80% RH (sin condensación)', '32.60', 'MEMFLA'),  
(('ZENMP48GB300', NULL, 'Creative Zen MP4 8GB Style 300', 'Características:\r\n\r\n\r\n8 GB de capacidad\r\n\r\nAutonomía: 32 horas con archivos MP3 a 128 kbps\r\n\r\nPantalla TFT de 1,  
8 pulgadas y 64.000 colores\r\n\r\nFormatos de audio compatibles: MP3, WMA (DRM9), formato Audible 4\r\n\r\nFormatos de foto compatibles: JPEG (BMP, TIFF, GIF y PNG\r\n\r\nFormatos de  
vídeo compatibles: AVI transcodificado (Motion JPEG)\r\n\r\n\r\nEqualizador de 5 bandas con 8 preajustes\r\n\r\nMicrófono integrado para grabar voz\r\n\r\nAltavoz y radio FM incorporada',  
'58.90', 'MP3');";  
  
//ejecutamos la sentencia sql  
$db->exec($sql);
```

Debemos tener cuidado en esta parte porque en la descripción de los productos hay comillas dobles (“) que nos darán problemas. Hay que ponerles una barra (\) para que no interfieran, quedarían así (\”).

o: 2.5 \", 250 GB, 5400 RPM, \r\

Creamos la tabla – Stock –.

```
/*----- TABLA STOCK -----*/
//sentencia SQL para crear la tabla STOCK
$sql= "CREATE TABLE IF NOT EXISTS `stock` (
`producto` varchar(12) COLLATE utf8_spanish_ci NOT NULL,
`tienda` int(11) NOT NULL,
`unidades` int(11) NOT NULL,
PRIMARY KEY (`producto`,`tienda`),
KEY `stock_ibfk_2` (`tienda`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_spanish_ci;";
//ejecutamos la sentencia
$db->exec($sql);
```

Rellenamos con la información necesaria. Este fragmento de código no se ve entero pues es muy largo.

```
//Volcar la base de datos para la tabla `familia`
$sql="INSERT INTO `stock` (`producto`, `tienda`, `unidades`) VALUES
('3DSNG', 1, 1),
('3DSNG', 2, 1),
('3DSNG', 3, 1),
('ACERAX3950', 1, 1),
('ARCLPMP32GBN', 2, 1),
('ARCLPMP32GBN', 3, 2),
('BRAVIA2BX400', 3, 1),
('EEEP1005PXD', 1, 2),
('EEEP1005PXD', 2, 1),
('HPMIN1103120', 2, 1),
('HPMIN1103120', 3, 2),
('IXUS115HSAZ', 2, 2),
('KSTDT101G2', 3, 1),
('KSTDTG332GBR', 2, 2),
('KSTMSDHC8GB', 1, 1),
('KSTMSDHC8GB', 2, 2),
('KSTMSDHC8GB', 3, 2),
('LEGRIAFS306', 2, 1),
('LGM237WDP', 1, 1),
```

```
('PIXMAMP252', 2, 1),
('PS3320GB', 1, 1),
('PWSHTA3100PT', 2, 2),
('PWSHTA3100PT', 3, 2),
('MSGCLX3175', 2, 1),
('SMN150101LD', 3, 1),
('SMSSMXC200PB', 2, 1),
('STYLUSX515W', 1, 1),
('TSSD16GBC10J', 3, 2),
('ZENMP48GB300', 1, 3),
('ZENMP48GB300', 2, 2),
('ZENMP48GB300', 3, 2);";
//ejecutamos la sentencia sql
$db->exec($sql);
/*----- FIN TABLA STOCK -----*/
```

Creamos la tabla – Tienda–. Y la rellenamos.

```
/*----- TABLA TIENDA -----*/
//sentencia SQL para crear la tabla TIENDA
$sql= "CREATE TABLE IF NOT EXISTS `tienda` (
`cod` int(11) NOT NULL AUTO_INCREMENT,
`nombre` varchar(100) COLLATE utf8_spanish_ci NOT NULL,
`tlf` varchar(13) COLLATE utf8_spanish_ci DEFAULT NULL,
PRIMARY KEY (`cod`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_spanish_ci AUTO_INCREMENT=4 ;";
//ejecutamos la sentencia
$db->exec($sql);

// Volcar la base de datos para la tabla `tienda`
$sql="INSERT INTO `tienda` (`cod`, `nombre`, `tlf`) VALUES
(1, 'CENTRAL', '600100100'),
(2, 'SUCURSAL1', '600100200'),
(3, 'SUCURSAL2', NULL);";
//ejecutamos la sentencia sql
$db->exec($sql);
/*----- FIN TABLA TIENDA -----*/
```



```
// Volcar la base de datos para la tabla `tienda`
$sql="INSERT INTO `tienda` (`cod`, `nombre`, `tlf`) VALUES
(1, 'CENTRAL', '600100100'),
(2, 'SUCURSAL1', '600100200'),
(3, 'SUCURSAL2', NULL);";
//ejecutamos la sentencia sql
$db->exec($sql);

/*----- FIN TABLA TIENDA -----*/
```

Creamos la tabla – Usuarios–. Y la rellenamos.

```
/*----- TABLA USUARIOS -----*/
//sentencia SQL para crear la tabla USUARIOS
$sql= " CREATE TABLE IF NOT EXISTS `usuarios` (
  `usuario` varchar(20) COLLATE utf8_spanish_ci NOT NULL,
  `contrasena` varchar(32) COLLATE utf8_spanish_ci NOT NULL,
  PRIMARY KEY (`usuario`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_spanish_ci;";
//ejecutamos la sentencia
$db->exec($sql);

// Volcar la base de datos para la tabla `usuarios`
$sql=" INSERT INTO `usuarios` (`usuario`, `contrasena`) VALUES
('dwes', 'e8dc8ccd5e5f9e3a54f07350ce8a2d3d');";
//ejecutamos la sentencia sql
$db->exec($sql);

/*----- FIN TABLA USUARIOS -----*/
```

Creamos la tabla – Ordenadores–. Y la rellenamos con la información necesaria para realizar el ejercicio.

```
/*----- TABLA ORDENADOR -----*/
//sentencia SQL para crear la tabla ORDENADOR
$sql= "CREATE TABLE IF NOT EXISTS `ordenador` (
  `cod` varchar(12) COLLATE utf8_spanish_ci NOT NULL,
  `procesador` varchar(50) COLLATE utf8_spanish_ci NOT NULL,
  `RAM` int(11) NOT NULL COMMENT 'En GB',
  `disco` varchar(50) COLLATE utf8_spanish_ci NOT NULL COMMENT 'Indicar numero, tecnología y capacidad',
  `grafica` varchar(50) COLLATE utf8_spanish_ci NOT NULL,
  `unidadoptica` varchar(50) COLLATE utf8_spanish_ci NOT NULL,
  `SO` varchar(50) COLLATE utf8_spanish_ci NOT NULL,
  `otros` varchar(250) COLLATE utf8_spanish_ci DEFAULT NULL,
  PRIMARY KEY (`cod`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_spanish_ci;";
//ejecutamos la sentencia
$db->exec($sql);
```

```
// Insertamos los datos en la tabla `ordenador`
$sql=" INSERT INTO `ordenador` (`cod`, `procesador`, `RAM`, `disco`, `grafica`, `unidadoptica`, `SO`, `otros`) VALUES
('ACERAX3950', 'Intel Core i5-650', 4, '1 disco SATA2 1TB', 'Nvidia GT320 1GB', 'DVD+-R DL 16x', 'Windows 7 Home Premium', NULL),
('PBELLI810323', 'Intel Core i3-550', 4, '1 disco SATA2 640GB', 'Nvidia G210M D3 512MB', 'DVD+-R DL', 'Windows 7 Home Premium',
'Equipo integrado con pantalla táctil 16:9 HD 23");";
//ejecutamos la sentencia sql
$db->exec($sql);

/*----- FIN TABLA ORDENADORES -----*/
```

Y con esto la base de datos estaría creada y dispuesta para la tarea.

Creación y modificación del código

Hacemos un nuevo archivo llamado Ordenador.php que contendrá la clase Ordenador. Lo guardamos en la carpeta incluida ya que es donde se encuentra el resto de archivo del mismo tipo, los que no tienen un archivo .tpl asociado.

```
<?php
class Ordenador{

//PROPIEDADES
    private $nombre_corto;
    private $codigo;
    private $procesador;
    private $ram;
    private $grafica;
    private $unidadoptica;
    private $otros;
    private $PVP;
    private $descripcion;

    //GETTERS
    public function getnombrecorto(){return $this->nombre_corto;}
    public function getcodigo(){return $this->codigo;}
    public function getprocesador(){return $this->procesador;}
    public function getram(){ return $this->ram;}
    public function getgrafica(){return $this->grafica;}
    public function getunidadoptica(){ return $this->unidadoptica;}
    public function getotros(){return $this->otros;}
    public function getPVP(){return $this->PVP;}
    public function getdescripcion(){return $this->descripcion;}

    //CONSTRUCTOR
    public function __construct($row){
        $this->nombre_corto = $row['nombre_corto'];
        $this->codigo = $row['cod'];
        $this->procesador = $row['procesador'];
        $this->ram = $row['RAM'];
        $this->grafica = $row['grafica'];
        $this->unidadoptica = $row['unidadoptica'];
        $this->otros = $row['otros'];
        $this->PVP = $row['PVP'];
        $this->descripcion = $row['descripcion'];
    }

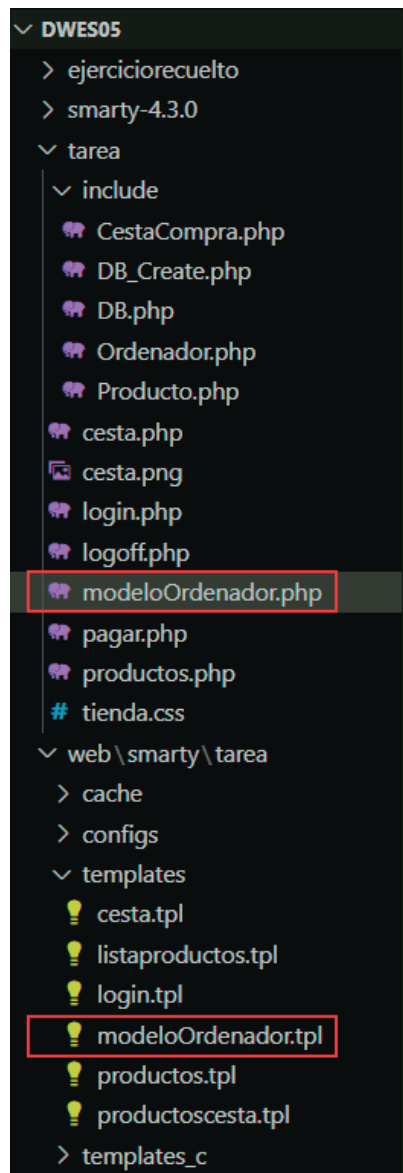
}

} //clase ORdenador
```

Para poder sacar la información necesaria de cada Ordenador debemos ir al archivo DB.php y crear una nueva función con la que realizaremos la búsqueda pertinente en la base de datos.

```
//Funcion para obtener los datos de los ordenadores
public static function obtieneOrdenador($codigo){
    //sentencia SQL para sacar los datos del ordenador
    $sql="SELECT ordenador.cod, nombre_corto, PVP, descripcion, procesador, RAM, grafica, unidadoptica, otros FROM ordenador INNER JOIN producto
    ON producto.cod = ordenador.cod where ordenador.cod='".$codigo."'";
    //llamamos a la funcion ejecutarConsulta que se encuentra más arriba en este mismo archivo
    $result=self::ejecutaConsulta($sql);
    //declaramos una varibale ordenador que nos servirá para crear posteriormente un objeto de la clase Ordenador.
    $ordenador = null;
    //sacamos los datos de la consulta SQL y los pasamos con la array que nos crea el fetch al constructor de la clase Ordenador
    if(isset($result)){
        //guardamos el array resultante con los datos sacados por la sentencia SQL
        $row = $result->fetch();
        //Se los pasamos al constructor de la clase para crear un objeto con estas características
        $ordenador = new Ordenador($row);
    }
    //devolvemos el objeto ordenador
    return $ordenador;
} //function obtener datos de ordenador
```

Ahora creamos un archivo .php, llamado modeloOrdenador.php, donde visualizaremos los datos sacados del Ordenador. También debemos crear el archivo .tpl asociado, con el mismo nombre y en la carpeta de los templates.



En modeloOrdenador.php debemos incluir las librerías del Smarty y crear una variable código que mandaremos a la función obtieneOrdenador del archivo DB.php para que esta nos devuelva un objeto Ordenador creado a través de la clase Ordenador.

```
<?php
//archivos requeridos
require_once('include/DB.php');
require_once('Smarty.class.php');

// Recuperamos la información de la sesión
session_start();

// Y comprobamos que el usuario se haya autenticado
if (!isset($_SESSION['usuario']))
    die("Error - debe <a href='login.php'>identificarse</a>.<br />");

// Cargamos la librería de Smarty
$smarty = new Smarty;
$smarty->template_dir = '../web/smarty/tarea/templates/';
$smarty->compile_dir = '../web/smarty/tarea/templates_c/';
$smarty->config_dir = '../web/smarty/tarea/configs/';
$smarty->cache_dir = '../web/smarty/tarea/cache/';

$smarty->assign('usuario', $_SESSION['usuario']);
//recogemos el valor del código desde la url
$smarty->assign('codigo', $_GET['codigo']);
//mandamos ese código a la sentencia SQL para realizar la búsqueda y poder mostrar los datos
$smarty->assign('ordenador', DB::obtieneOrdenador($_GET['codigo']));

// Mostramos la plantilla
$smarty->display('modeloOrdenador.tpl');

?>
```

En modeloOrdenador.tpl escribimos con html una plantilla que nos permita visualizar la información del Ordenador manteniendo una continuidad en el diseño.

```
<body class="pagcesta">
    <div id="contenedor">
        <div id="encabezado">
            {*NOMBRE CORTO DEL PRODUCTO*}
            <h1>{$ordenador->getnombrecorto()}</h1>
            {*CÓDIGO DEL PRODUCTO*}
            <p><strong>Código: </strong> {$ordenador->getcodigo()}</p>
        </div>
        <div id="productos">
            {*MOSTRAR LOS DATOS DEL ORDENADOR*}
            <h2>Características</h2>
            {*PROCESADOR*}
            <p><strong>Procesador: </strong> <span>{$ordenador->getprocesador()}</span> </p>
            {*RAM*}
            <p><strong>RAM: </strong> <span>{$ordenador->getram()}</span> </p>
            {*TARJETA GRÁFICA*}
            <p><strong>Tarjeta gráfica: </strong> <span>{$ordenador->getgrafica()}</span> </p>
            {*UNIDAD ÓPTICA*}
            <p><strong>Unidad óptica: </strong> <span>{$ordenador->getunidadoptica()}</span> </p>
            {*OTROS*}
            <p><strong>Otros: </strong> <span>{$ordenador->getotros()}</span> </p>
            {*PVP*}
            <p><strong>PVP: </strong> <span>{$ordenador->getPVP()}</span> </p>
            {*DESCRIPCIÓN*}
            <h2 class="mOrdenador">Descripción</h2>
            <p>{$ordenador->getdescripcion()}</p>
        </div>
        <div id="pie">
            {*BOTON PARA VOLVER A productos.php*}
            <button><a href="productos.php">Volver a Producto</a></button>
        </div>
    </div>
</body>
</html>
```

Y quedaría así si hemos hecho todo correctamente.

Acer AX3950 I5-650 4GB 1TB W7HP

Código: ACERAX3950

Características

Procesador: Intel Core i5-650

RAM: 4

Tarjeta gráfica: Nvidia GT320 1GB

Unidad óptica: DVD+-R DL 16x

Otros:

PVP: 410.00

Descripción

Características: Sistema Operativo : Windows® 7 Home Premium Original Procesador / Chipset Número de Ranuras PCI: 1 Fabricante de Procesador: Intel Tipo de Procesador: Core i5 Modelo de Procesador: i5-650 Núcleo de Procesador: Dual-core Velocidad de Procesador: 3,20 GHz Caché: 4 MB Velocidad de Bus: No aplicable Velocidad HyperTransport: No aplicable Interconexión QuickPathNo aplicable Procesamiento de 64 bits: Si Hyper-ThreadingSi Fabricante de Chipset: Intel Modelo de Chipset: H57 Express Memoria Memoria Estándar: 4 GB Memoria Máxima: 8 GB Tecnología de la Memoria: DDR3 SDRAM Estándar de Memoria: DDR3-1333/PC3-10600 Número de Ranuras de Memoria (Total): 4 Lector de tarjeta memoria: Si Soporte de Tarjeta de Memoria: Tarjeta CompactFlash (CF) Soporte de Tarjeta de Memoria: MultiMediaCard (MMC) Soporte de Tarjeta de Memoria: Micro Drive Soporte de Tarjeta de Memoria: Memory Stick PRO Soporte de Tarjeta de Memoria: Memory Stick PRO Soporte de Tarjeta de Memoria: CF+ Soporte de Tarjeta de Memoria: Tarjeta Secure Digital (SD) Storage Capacidad Total del Disco Duro: 1 TB RPM de Disco Duro: 5400 Tipo de Unidad Óptica: Grabadora DVD Compatibilidad de Dispositivo Óptico: DVD-RAM/±R/±RW Compatibilidad de Medios de Doble Capa: Si

[Volver a Producto](#)

La imagen del código muestra una versión final, durante el desarrollo de estos dos archivos se crea una variable con un código de producto de la bbdd y fui probando hasta que salió el resultado deseado.

Para pasar el valor por enlace como se ve en la versión final debemos abrir listaproductos.tpl y modificamos el código de manera que solo los productos cuyo código de Familia sea ORDENA sean un enlace.

listaproducto.tpl El original

```
{foreach from=$productos item=producto}
    <p><form id='{ $producto->getcodigo()}' action='productos.php' method='post'>
        <input type='hidden' name='cod' value='{ $producto->getcodigo()}' />
        <input type='submit' name='enviar' value='Añadir' />
        { $producto->getnombrecorto()}: { $producto->getPVP()} euros.</form></p>
{/foreach}
```

listaproducto.tpl Modificado

```
{foreach from=$productos item=producto}
    <p>
        <form id='{ $producto->getcodigo()}' action='productos.php' method='post'>
            <input type='hidden' name='cod' value='{ $producto->getcodigo()}' />
            <input type='submit' name='enviar' value='Añadir' />
{*-Modificación frente al original-*}
        {*Le damos valor a la variable código que luego pasaremos por url a modeloOrdenador.php*}
        { $codigo= $producto->getcodigo()}
        {*Si la familia del producto es ORDENA entonces se mostrará como enlace*}
        {if $producto->getfamilia() == "ORDENA"}
            {*Pasamos el valor del código por la url*}
            <a href='../tarea/modeloOrdenador.php?codigo={ $codigo}'>
                { $producto->getnombrecorto()}: { $producto->getPVP()} euros.
            </a>
        {*sino se mostrará de manera normal*}
        {else}
            { $producto->getnombrecorto()}: { $producto->getPVP()} euros.
        {/if}
{*- FIN Modificación frente al original-*}
    </form>
</p>
{/foreach}
```

Ahora debemos modificar las búsquedas en la base de datos sobre los productos para que saquen también la familia a al que pertenecen y esta distinción se válida cuando el código del producto no esté forzado.

Volvemos a DB.php y en la función que saca la información de los productos debemos añadirle familia a la sentencia sql.

```
public static function obtieneProducto($codigo) {
    //se ha añadido familia a la búsqueda para así separar los de tipo ordenador del resto
    $sql = "SELECT cod, nombre_corto, nombre, PVP, familia FROM producto";
    $sql .= " WHERE cod='" . $codigo . "'";
    $resultado = self::ejecutaConsulta ($sql);
    $producto = null;

    if(isset($resultado)) {
        $row = $resultado->fetch();
        $producto = new Producto($row);
    }

    return $producto;
}
// funcion obtiene producot por CODIGO
```

Ahora vamos a la clase Producto.php que también se encuentra en la carpeta include, y añadimos la propiedad familia a la lista de propiedades. Debemos hacer lo propio con el constructor y los getters.

```
<?php

class Producto {
    protected $codigo;
    protected $nombre;
    protected $nombre_corto;
    protected $PVP;
    //propiedad familia, os ayudará a separar los ordenadores de los que no lo son
    protected $familia;

    public function getcodigo() {return $this->codigo; }
    public function getnombre() {return $this->nombre; }
    public function getnombrecorto() {return $this->nombre_corto; }
    public function getPVP() {return $this->PVP; }
    //método get para la propiedad familia
    public function getfamilia() {return $this->familia; }

    public function __construct($row) {
        $this->codigo = $row['cod'];
        $this->nombre = $row['nombre'];
        $this->nombre_corto = $row['nombre_corto'];
        $this->PVP = $row['PVP'];
        //asignación del valor sacado de la array $row a la propiedad familia
        $this->familia = $row['familia'];
    }
}

//clase Producto

?>
```

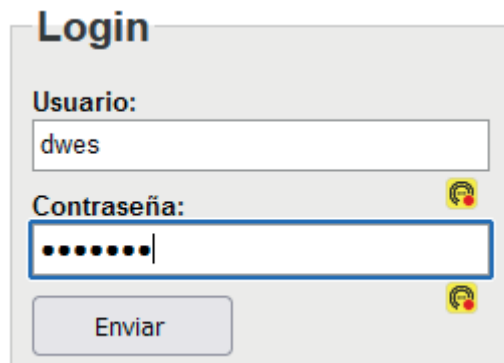

Llegamos a este punto es necesario comprobar el código sin forzar un código de producto, sino usar el que pasamos por url desde listaproductos.tpl y recogemos en modeloOrdenador con una variable GET.

```
{*-Modificación frente al original--*}
{*Le damos valor a la variable código que luego pasaremos por url a modeloOrdenador.php*}
{$codigo=$producto->getcódigo()}
{*Si la familia del producto es ORDENA entonces se mostrará como enlace*}
{if $producto->getfamilia() == "ORDENA"}
    {*Pasamos el valor del código por la url*}
    <a href=" ../tarea/modeloOrdenador.php?codigo={$codigo}">
        {$producto->getnombrecorto()}: {$producto->getPVP()} euros.
```

```
//recogemos el valor del código desde la url
$smarty->assign('codigo', $_GET['codigo']);
//mandamos ese código a la sentencia SQL para realizar la búsqueda y poder mostrar los datos
$smarty->assign('ordenador', DB::obtieneOrdenador($_GET['codigo']));
```

Comprobación final de la Tarea

Abrimos login.php e introducimos el usuario y contraseña



Login

Usuario:

Contraseña:

Como los datos han sido correctos debe llevarnos a productos.php, aquí nos mostrará el nombre de los productos. Solo los productos de la familia ordenador deben ser enlaces.

Listado de productos

Añadr

Nintendo 3DS negro: 270.00 euros.

Añadr

[Acer AX3950 I5-650 4GB 1TB W/HP: 410.00 euros.](#)

Añadr

Archos Clipper MP3 2GB negro: 26.70 euros.

Añadr

Sony Bravia 32IN FULLHD KDL-32BX400: 356.90 euros.

Añadr

Asus EEEPC 100SPXD N455 1 250 BL: 245.40 euros.

Añadr

HP Mini 110-3120 10.1LED N455 1GB 250GB W7S negro: 270.00 euros.

Añadr

Canon Ixus 115HS azul: 196.70 euros.

Añadr

Kingston DataTraveler 16GB DT101G2 USB2.0 negro: 19.20 euros.

Añadr

Kingston DataTraveler G3 32GB rojo: 40.00 euros.

Añadr

Kingston MicroSDHC 8GB: 10.20 euros.

Añadr

Canon Legria FS306 plata: 175.00 euros.

Añadr

LG TDT HD 23 M237WDP-PC FULL HD: 186.00 euros.

Añadr

HP Laserjet Pro Wifi P1102W: 99.90 euros.

Añadr

Pentax Optio LS1100: 104.80 euros.

Añadr

Lector ebooks Papyre6 con SD2GB + 500 ebooks: 205.50 euros.

Añadr

[Packard Bell I8103 23 I3-550 4G 640GB NVIDIAG210: 761.80 euros.](#)

Añadr

Canon Pixma IP4850: 97.30 euros.

Añadr

Canon Pixma MP252: 41.60 euros.

Añadr

PS3 con disco duro de 320GB: 380.00 euros.

Añadr

Canon Powershot A3100 plata: 101.40 euros.

Añadr

Samsung CLX3175: 190.00 euros.

Añadr

Samsung N150 10.1LED N450 1GB 250GB BAT6 BT W7 R: 260.60 euros.

Añadr

Samsung SMX-C200PB EDC ZOOM 10X: 127.20 euros.

Añadr

Epson Stylus SX515W: 77.50 euros.

Añadr

Toshiba SD16GB Class10 Jewel Case: 32.60 euros.

Añadr

Creative Zen MP4 8GB Style 300: 58.90 euros.

Desconectar usuario dwes

Cesta

Cesta vacía

Vaciar Cesta

Comprar

Hacemos click en los dos y comprobamos que sale la información de manera correcta. Además se ha añadido un botón de regreso a productos.php para que sea más cómoda la navegación.

Acer AX3950 I5-650 4GB 1TB W7HP

Código: ACERAX3950

Carasterísticas

Procesador: Intel Core i5-650

RAM: 4

Tarjeta gráfica: Nvidia GT320 1GB

Unidad óptica: DVD+-R DL 16x

Otros:

PVP: 410.00

Descripción

Características: Sistema Operativo : Windows® 7 Home Premium Original Procesador / Chipset Número de Ranuras PCI: 1 Fabricante de Procesador: Intel Tipo de Procesador: Core i5 Modelo de Procesador: i5-650 Núcleo de Procesador: Dual-core Velocidad de Procesador: 3,20 GHz Caché: 4 MB Velocidad de Bus: No aplicable Velocidad HyperTransport: No aplicable Interconexión QuickPathNo aplicable Procesamiento de 64 bits: Si Hyper-ThreadingSi Fabricante de Chipset: Intel Modelo de Chipset: H57 Express Memoria Memoria Estándar: 4 GB Memoria Máxima: 8 GB Tecnología de la Memoria: DDR3 SDRAM Estándar de Memoria: DDR3-1333/PC3-10600 Número de Ranuras de Memoria (Total): 4 Lector de tarjeta memoria: Si Soporte de Tarjeta de Memoria: Tarjeta CompactFlash (CF) Soporte de Tarjeta de Memoria: MultiMediaCard (MMC) Soporte de Tarjeta de Memoria: Micro Drive Soporte de Tarjeta de Memoria: Memory Stick PRO Soporte de Tarjeta de Memoria: Memory Stick Soporte de Tarjeta de Memoria: CF+ Soporte de Tarjeta de Memoria: Tarjeta Secure Digital (SD) Storage Capacidad Total del Disco Duro: 1 TB RPM de Disco Duro: 5400 Tipo de Unidad Óptica: Grabadora DVD Compatibilidad de Dispositivo Óptico: DVD-RAM/±R/±RW Compatibilidad de Medios de Doble Capa: Si

Volver a Producto

Packard Bell I8103 23 I3-550 4G 640GB NVIDIAG210

Código: PBELLI810323

Carasterísticas

Procesador: Intel Core i3-550

RAM: 4

Tarjeta gráfica: Nvidia G210M D3 512MB

Unidad óptica: DVD+-R DL

Otros: Equipo integrado con pantalla táctil 16:9 HD 23"

PVP: 761.80

Descripción

Características: CPU CHIPSET Procesador : G3-550 NorthBridge : Intel H57 MEMORIA Memoria Rma : Ddr3 4096 MB DISPOSITIVOS DE ALMACENAMIENTO Disco Duro: 640Gb 7200 rpm Óptico : Slot Load siper multi Dvdw Lector de Tarjetas: 4 in 1 (XD, SD, HC, MS, MS PRO, MMC) dispositivos gráficos Monitor: 23" HD Tarjeta Gráfica: Nvidia G210M D3 512Mb Memoria Máxima: Hasta 1918Mb AUDIO Audio Out: 5.1 Audio Out Audio In: 1 jack Heasphone in: 1x jack Altavoces: Stereo ACCESORIOS Teclado: Teclado y ratón inalámbrico Mando a distancia: EMEA Win7 WMC COMUNICACIONES Wireless: 802.11 b/g/n mini card Tarjeta de Red: 10/100/1000 Mbps Bluetooth: Bluetooth Webcam: 1Mpíxel Hd (1280x720) Tv tuner: mCARD/SW/ DVB-T MONITOR Tamaño: 23" contraste: 1000:1 Tiempo de respuesta: 5MS Resolución: 1920 X 1080 PUERTOS E/S Usb 2.0 : 6 Mini Pci-e : 2 Esata: 1 SISTEMA OPERATIVO O.S: Microsoft Windows 7 Premium

Volver a Producto