PHP Avanzado

Módulo 06 - Workshop

Para realizar los workshops se recomienda...

- Leer los contenidos previos.
- Descargar el material necesario de la sección de descargas.

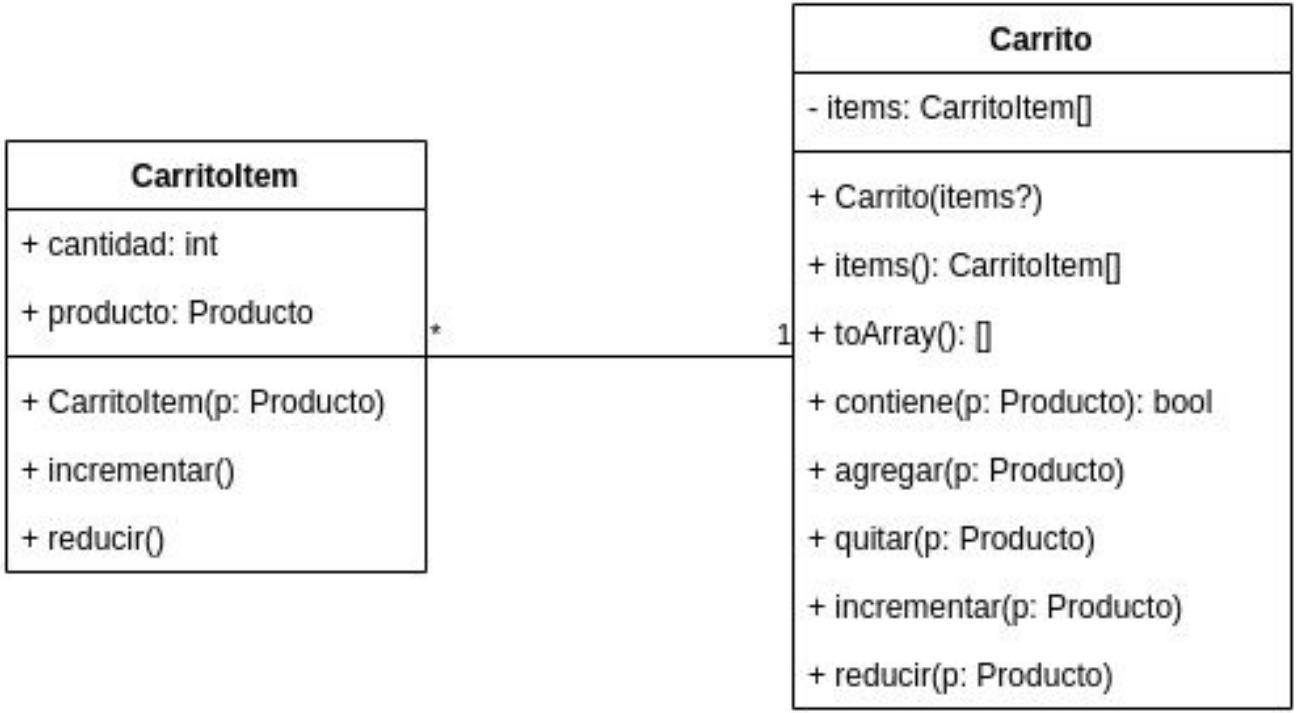
Workshop

Para hacer en clase

¡Otro requerimiento bien intenso nos toca! Vamos a ello, que se está poniendo divertido. Esta vez, vamos a dividir los requerimientos según la capa a modificar. ¿Qué es esto? Vamos a dividir los requerimientos según sea modificar el núcleo de la aplicación (core), las interfaces de la aplicación (app) o el Front Controller (public)

Requerimientos de la capa Central (core).

Aprovechando la gran cantidad de herramientas que tenemos, vamos a completar las estructuras de datos fijas de nuestra aplicación. Nuestra aplicación es una tienda online, por lo que debemos crear las estructuras de datos necesarias para ello. Un carrito puede ser pensado como array, y una excelente forma de diseñar una estructura de array con clases es diseñar una clase para la lista y otra clase para el item. Es decir, diseñar lo siguiente:


CarritoItem
+ cantidad: int
+ producto: Producto
+ CarritoItem(p: Producto)
+ incrementar()
+ reducir()
*
1
Carrito
- items: CarritoItem[]
+ Carrito(items?)
+ items(): CarritoItem[]
+ toArray(): []
+ contiene(p: Producto): bool
+ agregar(p: Producto)
+ quitar(p: Producto)
+ incrementar(p: Producto)
+ reducir(p: Producto)

Requerimientos de la capa de Aplicación

Acá es donde tenemos el mayor trabajo que hacer. Así que vamos por partes. Recordemos que la capa de Aplicación tiene las distintas entradas y salidas externas, así que es donde ocurre mucho código.

- El Carrito será un objeto de la clase Carrito compartido a lo largo de varias peticiones HTTP. Podríamos pensar en un middleware Carrito. Sin embargo, los Middleware son compartidos a lo largo de una sola petición HTTP. El Carrito necesita ser compartido entre varias peticiones HTTP.
- Para eso nos sirven las variables de sesión. Podemos crear un Middleware que haga un session_start(), así podremos usar las variables de sesión para compartir el carrito.
- Una funcionalidad muy interesante es el envío de mails. Así que deberemos agregar una nueva carpeta dentro de app para tener todos los conectores referentes al envío de mails. El envío de mail es una excelente oportunidad para practicar la Inyección de Dependencias (D) ¿Por qué? Porque para enviar un mail necesitamos llamar a la función de envío de mails nativa de PHP. Ahora bien, esa función es muy limitada, por lo que se usa, en producción, la librería PHP Mailer.

- También es necesario contemplar este caso: al usar PHP Mailer estamos utilizando servidores SMTP y es posible que el hosting donde alojemos el proyecto necesite configurar determinados parámetros. Es por esa razón que, muchas veces, el propio proveedor de hosting nos facilita su versión de PHP Mailer adaptada a su configuración (me ha pasado). Todo esto nos pone delante de algo: deberíamos poder cambiar fácilmente de mailer sin cambiar todo el código. Esa es exactamente la utilidad de los abstractos, de la inyección de dependencias y de Liskov.

- Tendríamos que agregar una vista nueva donde se muestre el estado actual del carrito, a la vez que debemos modificar la página producto para agregar el botón para agregar al carrito. En el ejemplo final tienen una versión muy pobre de la interfaz. He preferido, sinceramente, no involucrarme más de lo estrictamente necesario con el front-end ya que tenemos muchas cosas nuevas del back. Pero te animo a que hagas mejores diseños que el que ofrezco acá.

Requerimientos del Front Controller (public)

Acá también tenemos varios cambios:

- El primer gran cambio que te propongo es empezar a usar la autocarga de clases y los namespaces de PHP. Así evitamos tantos require innecesarios. Recuerda que una buena práctica es diseñar los namespaces siguiendo la misma estructura que tus archivos y carpetas, así el algoritmo del spl_autoload_register que diseñemos será mucho más breve.
- El segundo requerimiento es que actualices las rutas y los middlewares. Hay que incluir el Session. Hay que inyectar el Mailer al igual que inyectamos la conexión a la base de datos. Hay que agregar las rutas para agregar al carrito, ver pedido y enviar el mail con el pedido

¡Muchas gracias!

¡Sigamos trabajando!