Presentación

Nombre: Andy Manuel

Apellidos: Reyes Del Rosario

Matricula: 100488534

Carrera: Licenciatura en Informática

Asignatura: INF5170

Sección: 01

Maestro: Radhames Silverio González

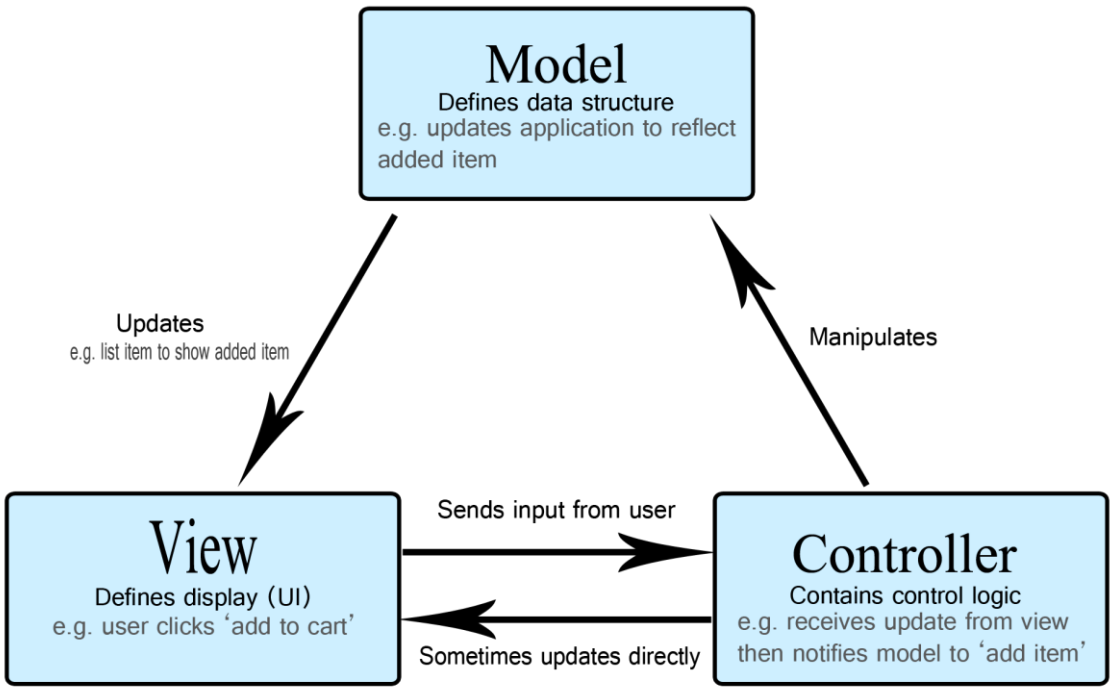
MVC

MVC (Modelo-Vista-Controlador) es un patrón en el diseño de software comúnmente utilizado para implementar interfaces de usuario, datos y lógica de control. Enfatiza una separación entre la lógica de negocios y su visualización. Esta "separación de preocupaciones" proporciona una mejor división del trabajo y una mejora de mantenimiento. Algunos otros patrones de diseño se basan en MVC, como MVVM (Modelo-Vista-modelo de vista), MVP (Modelo-Vista-Presentador) y MVW (Modelo-Vista-Whatever).

Las tres partes del patrón de diseño de software MVC se pueden describir de la siguiente manera:

- Modelo: Maneja datos y lógica de negocios.
- Vista: Se encarga del diseño y presentación.
- Controlador: Enruta comandos a los modelos y vistas.

Imagine una sencilla aplicación de lista de compras. Todo lo que queremos es una lista del nombre, la cantidad y el precio de cada artículo que necesitamos comprar esta semana. A continuación, describiremos cómo podríamos implementar parte de esta funcionalidad usando MVC.



Modelo

El modelo define qué datos debe contener la aplicación. Si el estado de estos datos cambia, el modelo generalmente notificará a la vista (para que la pantalla pueda cambiar según sea necesario) y, a veces, el controlador (si se necesita una lógica diferente para controlar la vista actualizada).

Volviendo a nuestra aplicación de lista de compras, el modelo especificará qué datos deben contener los artículos de la lista (artículo, precio, etc.) y qué artículos de la lista ya están presentes.

Vista

La vista define cómo se deben mostrar los datos de la aplicación.

En nuestra aplicación de lista de compras, la vista definiría cómo se presenta la lista al usuario y recibiría los datos para mostrar desde el modelo.

Controlador

El controlador contiene una lógica que actualiza el modelo y / o vista en respuesta a las entradas de los usuarios de la aplicación.

Entonces, por ejemplo, nuestra lista de compras podría tener formularios de entrada y botones que nos permitan agregar o eliminar artículos. Estas acciones requieren que se actualice el modelo, por lo que la entrada se envía al controlador, que luego manipula el modelo según corresponda, que luego envía datos actualizados a la vista.

Sin embargo, es posible que también se desee actualizar la vista para mostrar los datos en un formato diferente, por ejemplo, cambiar el orden de los artículos de menor a mayor precio o en orden alfabético. En este caso, el controlador podría manejar esto directamente sin necesidad de actualizar el modelo.

MVC en la web

Como desarrollador web, este patrón probablemente será bastante familiar, incluso si nunca lo has usado conscientemente antes. Su modelo de datos probablemente esté contenido en algún tipo de base de datos (ya sea una base de datos tradicional del lado del servidor como MySQL, o una solución del lado del cliente como IndexedDB). El código de control de su aplicación probablemente esté escrito en HTML / JavaScript , y su interfaz de usuario probablemente esté escrita usando HTML / CSS / o lo que sea. Esto se parece mucho a MVC, pero MVC hace que estos componentes sigan un patrón más rígido.

En los primeros días de la Web, la arquitectura MVC se implementó principalmente en el lado del servidor, con el cliente solicitando actualizaciones a través de formularios o enlaces, y recibiendo vistas actualizadas para mostrar en el navegador. Sin embargo, en

estos días, mucha de la lógica se enviaba al cliente con la llegada de los almacenes de datos del lado del cliente, y XMLHttpRequest permitía actualizaciones parciales de la página según era necesario.

Los frameworks de desarrollo web como AngularJS y Ember.js implementan una arquitectura MVC, aunque de maneras ligeramente diferentes.