Частное учреждение образования

«Колледж бизнеса и права»

УТВЕРЖДАЮ

Ведущий

методист колледжа

\_\_\_\_\_\_\_\_\_\_\_ Е.В. Паскал

«\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_2021

|  |  |
| --- | --- |
| Специальность: 2-40 01 01 «Программное обеспечение информационных технологий» | Дисциплина: «Технология разработки программного обеспечения» |

**Лабораторная работа № 24**

**Инструкционно-технологическая карта**

Тема: Создание программной системы, использующей шаблон MVC

Цель: Научиться создавать программные системы, используя шаблон MVC

Время выполнения: 2 часа

**Контрольныевопросы**

1. Опишите шаблон MVC

**Теоретические сведения для выполнения работы**

MVC — это шаблон программирования, который позволяет разделить логику приложения на три части:

Model (модель). Получает данные от контроллера, выполняет необходимые операции и передаёт их в вид.

View (вид или представление). Получает данные от модели и выводит их для пользователя.

Controller (контроллер). Обрабатывает действия пользователя, проверяет полученные данные и передаёт их модели.

**Как работает MVC**

Лучше всего понять концепцию MVC можно на реальном примере — ресторане с фастфудом. В нём посетители (пользователи) подходят к кассиру (одновременно вид и контроллер), видят меню и заказывают какое-нибудь блюдо.

Кассир проверяет, всё ли в порядке с заказом, и после оплаты передаёт нужные данные повару (модель). Повар готовит заказанное блюдо, хотя понятия не имеет о том, как выглядит посетитель, оплатил ли он заказ и так далее.

Когда модель закончит свою работу, она отправит результат в вид — обратно кассиру, который, в свою очередь, отдаст готовое блюдо посетителю.

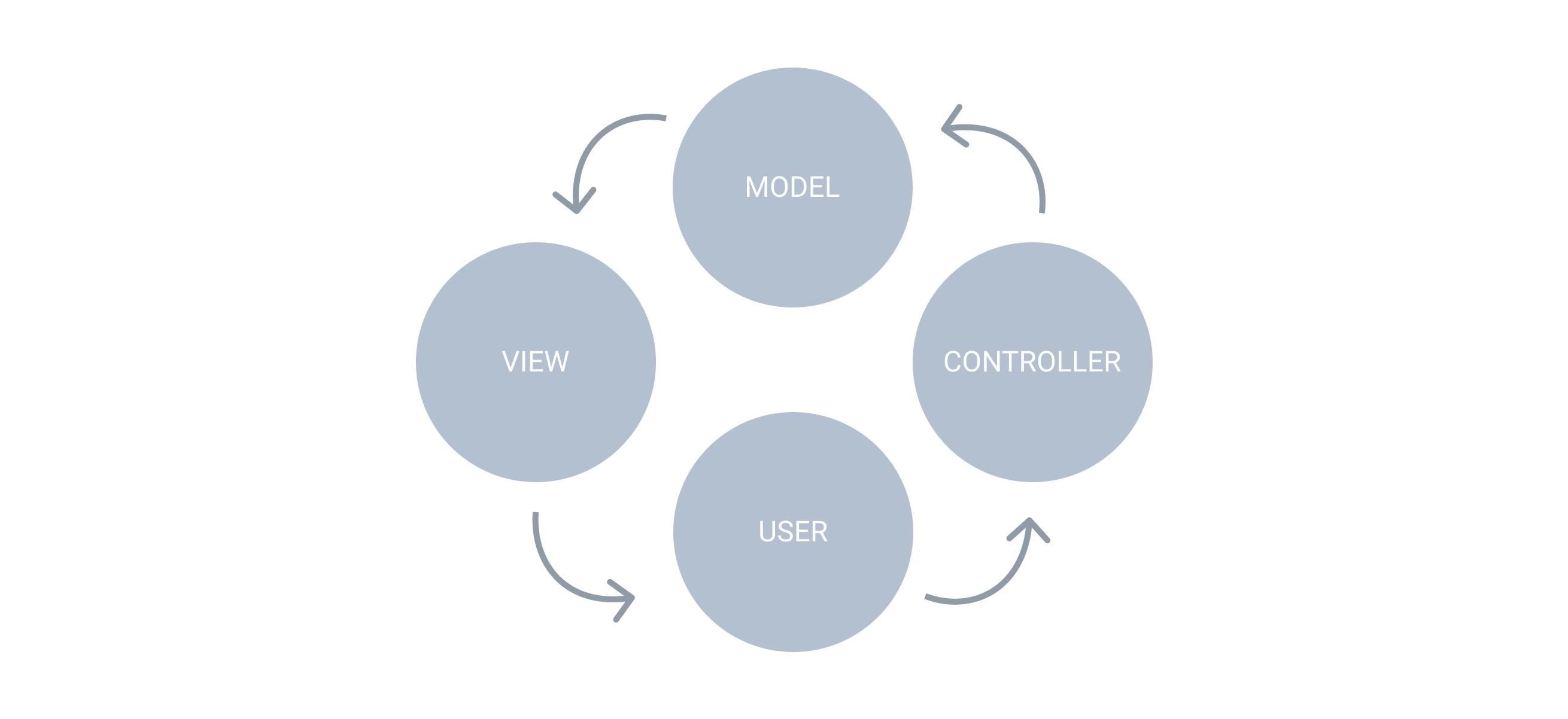
Если же говорить о приложениях, то компоненты будут следующие:

Вид — интерфейс.

Контроллер — обработчик событий, инициируемых пользователем (нажатие на кнопку, переход по ссылке, отправка формы).

Модель — метод, который запускается обработчиком и выполняет все основные операции (получение записей из базы данных, проведение вычислений).

Стоит также отметить, что реализация паттерна MVC может отличаться в зависимости от задачи. Например, в веб-разработке модель и вид взаимодействуют друг с другом через контроллер (как в примере с рестораном), а в приложениях модель может сама уведомлять вид, что нужно что-то изменить.



**Зачем программистам нужен MVC**

Этот паттерн разработки нужен для того, чтобы разделить логические части приложения и создавать их отдельно друг от друга. То есть писать независимые блоки кода, которые можно как угодно менять, не затрагивая другие.

Например, чтобы можно было переписать способ обработки данных, не меняя при этом способ их отображения. Это позволяет эффективно работать нескольким программистам — каждый занимается своим компонентом. При этом разработчику не нужно вникать в чужой код и его действия никак не повлияют на другие фрагменты приложения.

Независимым программистам MVC тоже будет полезен, потому что они смогут сосредоточиться на разработке одного компонента за раз.

Практика: пишем MVC-приложение

Чтобы лучше вникнуть в этот паттерн, стоит применить его на практике. Для этого создайте WPF-приложение и сверстайте такую форму:

<StackPanel VerticalAlignment="Center" HorizontalAlignment="Center">

<StackPanel Orientation="Horizontal">

<TextBox Name="Num1TextBox" Margin="5" Width="50"/>

<TextBlock Text="+" Margin="5" FontWeight="Bold"/>

<TextBox Name="Num2TextBox" Margin="5" Width="50"/>

</StackPanel>

<TextBlock Text="0" Name="ResultTextBlock" TextAlignment="Center" Margin="5"/>

<Button Content="Посчитать" Name="CalculateButton" Click="CalculateButton\_Click" Margin="5"/>

</StackPanel>

Это и есть View — его видит пользователь. Тут есть кнопка, при нажатии на которую вызывается Controller:

private void CalculateButton\_Click(object sender, RoutedEventArgs e) //Метод, который вызывается при нажатии на кнопку "Посчитать"

{

//Валидация полученных данных

string text1 = Num1TextBox.Text.Trim();

string text2 = Num2TextBox.Text.Trim();

int num1 = 0;

int num2 = 0;

if (!string.IsNullOrEmpty(text1) && !string.IsNullOrEmpty(text2))

{

try

{

num1 = Convert.ToInt32(text1);

num2 = Convert.ToInt32(text2);

}

catch (Exception exc) { }

Calculate(num1, num2); //Передача данных модели

}

}

Контроллер получает пользовательский ввод и обрабатывает данные. Он также может проверять права пользователя. Если валидация проходит успешно, данные передаются в Model:

public void Calculate(int num1, int num2)

{

result = num1 + num2; //Проведение операций с полученными данными

UpdateView(); //Вызов обновления представления

}

Модель проводит с этими данными необходимые операции, а затем вызывает метод обновления вида:

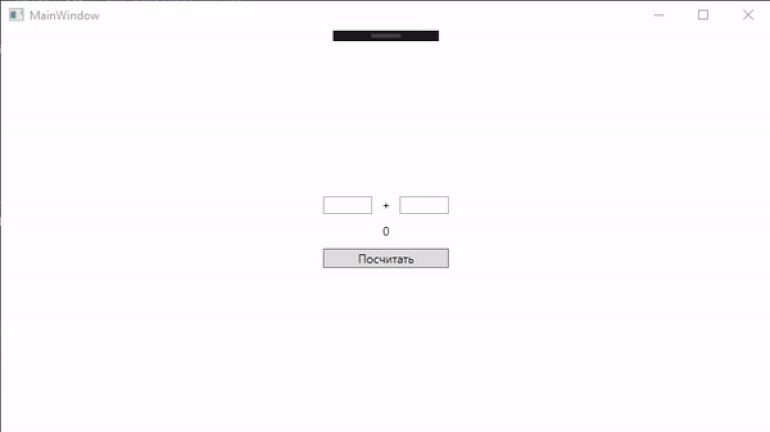
public void UpdateView()

{

ResultTextBlock.Text = result.ToString(); //Изменение вида

}

Вот результат работы такой программы:



**Порядок выполнения работы**

1. Изучите теоретические сведения.

2. Реализовать программу использующую MVC по теме индивидуального задания

Преподаватель \_\_\_\_\_\_\_\_ А.С. Комаровский

Рассмотрено на заседании цикловой

комиссии ПОИТ №10

Протокол № от «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2021

Председатель ЦК \_\_\_\_\_\_\_\_ В.Ю. Михалевич