Namespace Payment.Domain

Classes

CashPaymentStrategy

Represents a payment strategy that handles cash payments. Implements the IPaymentStrategy interface.

CreditCardPaymentStrategy

Represents a payment strategy that handles credit card payments. Implements the IPaymentStrategy interface.

DebitCardPaymentStrategy

Represents a payment strategy that handles debit card payments. Implements the IPaymentStrategy interface.

PaymentContext

Defines the payment context that uses a strategy for processing payments. The context allows the client to choose a payment strategy at runtime and delegates the payment process to the selected strategy.

PaymentService

Defines the service for processing payments. This class utilizes the Strategy pattern to select a payment method based on the selected payment type at runtime.

Interfaces

IPaymentStrategy

Defines the contract for payment strategies. Any payment method must implement this interface to define its specific payment behavior.

Enums

PaymentType

Defines the payment types available for processing.

Class CashPaymentStrategy

Namespace: Payment.Domain

Assembly: Payment.Domain.dll

Represents a payment strategy that handles cash payments. Implements the IPaymentStrategy interface.

```
public class CashPaymentStrategy : IPaymentStrategy
```

Inheritance

object ← CashPaymentStrategy

Implements

IPaymentStrategy

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

Pay(double)

Executes the payment process for cash payment.

```
public bool Pay(double amount)
```

Parameters

`amount` double

The amount to be paid.

Returns

bool

Returns `true` to indicate the payment was successful.

Class CreditCardPaymentStrategy

Namespace: Payment.Domain

Assembly: Payment.Domain.dll

Represents a payment strategy that handles credit card payments. Implements the IPaymentStrategy interface.

```
public class CreditCardPaymentStrategy : IPaymentStrategy
```

Inheritance

object ← CreditCardPaymentStrategy

Implements

IPaymentStrategy

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

Pay(double)

Executes the payment process for credit card payment.

```
public bool Pay(double amount)
```

Parameters

`amount` double

The amount to be paid.

Returns

bool

Returns `true` to indicate the payment was successful.

Class DebitCardPaymentStrategy

Namespace: Payment.Domain

Assembly: Payment.Domain.dll

Represents a payment strategy that handles debit card payments. Implements the IPaymentStrategy interface.

```
public class DebitCardPaymentStrategy : IPaymentStrategy
```

Inheritance

object ← DebitCardPaymentStrategy

Implements

IPaymentStrategy

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

Pay(double)

Executes the payment process for debit card payment.

```
public bool Pay(double amount)
```

Parameters

`amount` double

The amount to be paid.

Returns

bool

Returns `true` to indicate the payment was successful.

Interface IPaymentStrategy

Namespace: Payment.Domain

Assembly: Payment.Domain.dll

Defines the contract for payment strategies. Any payment method must implement this interface to define its specific payment behavior.

```
public interface IPaymentStrategy
```

Methods

Pay(double)

Executes the payment process.

```
bool Pay(double amount)
```

Parameters

`amount` double

The amount to be paid.

Returns

bool

Returns `true` if the payment was successful, otherwise `false`.

Class PaymentContext

Namespace: Payment.Domain

Assembly: Payment.Domain.dll

Defines the payment context that uses a strategy for processing payments. The context allows the client to choose a payment strategy at runtime and delegates the payment process to the selected strategy.

```
public class PaymentContext
```

Inheritance

object ← PaymentContext

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

Pay(double)

Executes the payment process by delegating the work to the strategy object. The context does not implement the payment algorithms itself.

```
public bool Pay(double amount)
```

Parameters

`amount` double

The amount to be paid.

Returns

bool

`true` if the payment was successful, otherwise `false`.

SetPaymentStrategy(IPaymentStrategy)

Sets the payment strategy at runtime. The client can choose which payment method to use by calling this method.

```
public void SetPaymentStrategy(IPaymentStrategy strategy)
```

Parameters

`strategy` IPaymentStrategy

The payment strategy to use.

Class PaymentService

Namespace: Payment.Domain

Assembly: Payment.Domain.dll

Defines the service for processing payments. This class utilizes the Strategy pattern to select a payment method based on the selected payment type at runtime.

```
public class PaymentService
```

Inheritance

object ← PaymentService

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

ProcessPayment(int, double)

Processes the payment based on the selected payment type. It delegates the payment execution to the corresponding strategy (CreditCard, DebitCard, or Cash).

```
public bool ProcessPayment(int SelectedPaymentType, double Amount)
```

Parameters

`SelectedPaymentType` int

The payment method selected by the user.

`Amount` double

The amount to be paid.

Returns

bool

`true` if the payment was successfully processed, otherwise `false`.

Exceptions

ArgumentException

Thrown when an invalid payment type is selected.

Enum PaymentType

Namespace: Payment.Domain

Assembly: Payment.Domain.dll

Defines the payment types available for processing.

```
public enum PaymentType
```

Fields

`Cash = 3`

Cash payment method.

`CreditCard = 1`

CreditCard payment method.

`DebitCard = 2`

DebitCard payment method.

Namespace Payment.Domain.Tests

Classes

PaymentTests

Unit tests for the PaymentService class. The tests validate the functionality of processing payments using different payment methods.

Class PaymentTests

Namespace: Payment.Domain.Tests

Assembly: Payment.Domain.Tests.dll

Unit tests for the PaymentService class. The tests validate the functionality of processing payments using different payment methods.

```
public class PaymentTests
```

Inheritance

object ← PaymentTests

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

GivenAValidPaymentTypeAndAmount_WhenProcessPayment_ResultIsSuccesful(int, double)

Tests if a valid payment type and amount result in a successful payment process.

```
[TestCase(1, 1000)]
[TestCase(2, 2000)]
[TestCase(3, 3000)]
public void GivenAValidPaymentTypeAndAmount_WhenProcessPayment_ResultIsSuccesful(int
paymentType, double amount)
```

Parameters

`paymentType` int

The payment type to be used (1 for CreditCard, 2 for DebitCard, 3 for Cash).

`amount` double

The amount to be processed.

GivenAnUnknownPaymentTypeAndAmount_WhenProcessPayment_ResultIsError(int, double)

Tests if an invalid payment type results in an error.

```
[TestCase(4, 4000)]
public void GivenAnUnknownPaymentTypeAndAmount_WhenProcessPayment_ResultIsError(int
paymentType, double amount)
```

Parameters

`paymentType` int

The invalid payment type.

`amount` double

The amount to be processed.