[](https://ida.interchain.io/)

[Interchain Developer Academy](https://ida.interchain.io/)/[Interchain Developer Academy](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)



Search

[Interchain Developer Academy](https://ida.interchain.io/)[Interchain Developer Academy](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

Search



Filters

Interchain Developer Academy

[](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Week 0 - Getting Started](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Getting Started](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Blockchain 101](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Blockchain History](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Public and Managed Blockchains](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Consensus in Distributed Networks](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Cryptography](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Self-Assessment Quiz](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Go Introduction - First Steps](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Go Basics](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Go Interfaces](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Control Structures in Go](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Arrays and Slices in Go](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Standard Packages in Go](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Concurrency in Go](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Good-To-Know Dev Terms](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Docker Introduction](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Week 1 - Introduction to the Interchain](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Introduction to the Interchain](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Blockchain Technology and the Interchain](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[The Interchain Ecosystem](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Getting ATOM and Staking It](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[A Blockchain App Architecture](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Accounts](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Transactions](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Messages](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Modules](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Protobuf](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Multistore and Keepers](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[BaseApp](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Queries](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Events](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Context](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Testing](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Relaying with IBC](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Interchain Security](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Bridges](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Migrations](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Week 1 Quiz](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Week 2 - First Steps](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[First Steps](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Setup Your Work Environment](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Run a Node, API, and CLI](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Ignite CLI](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Exercise - Make a Checkers Blockchain](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Store Object](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Create Custom Messages](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Create and Save a Game Properly](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Add a Way to Make a Move](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Emit Game Information](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Record the Game Winner](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Week 2 Exercise](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Week 3 - Introduction to IBC and CosmJS](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Introduction to IBC and CosmJS](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[What is IBC?](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[IBC/TAO - Connections (OPTIONAL)](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[IBC/TAO - Channels (OPTIONAL)](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[IBC/TAO - Clients (OPTIONAL)](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[IBC Token Transfer](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Interchain Accounts (OPTIONAL)](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[IBC Middleware (OPTIONAL)](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Create a Custom IBC Middleware (OPTIONAL)](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Integrate IBC Middleware Into a Chain (OPTIONAL)](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[IBC Tooling](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[What is CosmJS?](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Your First CosmJS Actions](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Compose Complex Transactions](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Learn to Integrate Keplr](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Create Custom CosmJS Interfaces](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Week 4 - Ignite CLI and IBC Advanced](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Ignite CLI and IBC Advanced](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Keep an Up-To-Date Game Deadline](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Keep Track Of How Many Moves Have Been Played](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Put Your Games in Order](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Auto-Expiring Games](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Let Players Set a Wager](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Handle wager payments](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Integration tests](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Incentivize Players](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Help Find a Correct Move](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Play With Cross-Chain Tokens](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Understand IBC Denoms](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Go Relayer](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Hermes Relayer](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Week 5 - CosmJS Advanced](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[CosmJS Advanced](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Create Custom Objects](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Create Custom Messages](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Get an External GUI](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Integrate CosmJS and Keplr](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Backend Script for Game Indexing](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Week 6 - IBC Deep Dive](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[IBC Deep Dive](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[IBC Application Developer Introduction](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Make a Module IBC-Enabled](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Adding Packet and Acknowledgment Data](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Extend the Checkers Game With a Leaderboard](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Create a Leaderboard Chain](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Week 7 - From Code to MVP to Production and Migrations](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[From Code to MVP to Production and Migrations](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Run in Production](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Prepare the Software to Run](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Prepare a Validator and Keys](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Prepare Where the Node Starts](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Prepare and Connect to Other Nodes](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Configure, Run, and Set Up a Service](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Prepare and Do Migrations](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Simulate Production in Docker](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Tally Player Info After Production](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Add a Leaderboard as a Module](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Migrate the Leaderboard Module After Production](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Simulate a Migration in Docker](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Final Exam](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[What's Next?](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

[Continue Your Interchain Journey](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html)

Docs Version Switcher

On this page

[Channel handshake callbacks](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#channel-handshake-callbacks)

[Middleware version negotiation](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#middleware-version-negotiation)

[Capabilities](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#capabilities)

[Code snippets](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#code-snippets)

[Packet callbacks](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#packet-callbacks)

[Code snippets](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#code-snippets-2)

[ICS-4 wrappers](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#ics-4-wrappers)

[Capabilities](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#capabilities-2)

[Code snippets](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#code-snippets-3)

[#Copy link](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#create-a-custom-ibc-middleware) **Create a Custom IBC Middleware**

When developing a custom IBC application, one of the first things to do is to implement the IBCModule interface, as seen [previously](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/4-ibc-app-steps.html).

The interface can be found [here (opens new window)↗](https://github.com/cosmos/ibc-go/blob/main/modules/core/05-port/types/module.go).

Likewise, when developing IBC middleware, the Middleware interface should be implemented. It can be found in the same file as the IBCModule and appears as follows:



Copy

type Middleware interface {

IBCModule

ICS4Wrapper

}



Middleware must implement IBCModule to wrap communication from core IBC to the underlying application, and must implement ICS4Wrapper to wrap communication from the underlying application to core IBC.

You know IBCModule from the IBC custom app section. ICS4Wrapper is an interface like so:



Copy

// This is implemented by ICS4 and all middleware wrapping the base application.

// The base application will call `sendPacket` or `writeAcknowledgement` of the middleware directly above them

// which will call the next middleware until it reaches the core IBC handler.

type ICS4Wrapper interface {

SendPacket(ctx sdk.Context, chanCap \*capabilitytypes.Capability, packet exported.Packet) error

WriteAcknowledgement(ctx sdk.Context, chanCap \*capabilitytypes.Capability, packet exported.Packet, ack exported.Acknowledgement) error

GetAppVersion(ctx sdk.Context, portID, channelID string) (string, bool)

}

When developing custom middleware, you can implement these interfaces for a new IBCMiddleware that has access to its keeper and an underlying application. In the ibc\_middleware.go file:



Copy

import (

...

porttypes "github.com/cosmos/ibc-go/v5/modules/core/05-port/types"

...

)

var \_ porttypes.Middleware = &IBCMiddleware{}

type IBCMiddleware struct {

app porttypes.IBCModule

keeper keeper.Keeper //add a keeper for stateful middleware

}

// IBCMiddleware creates a new IBCMiddleware given the associated keeper and underlying application

func NewIBCMiddleware(app porttypes.IBCModule, k keeper.Keeper) IBCMiddleware {

return IBCMiddleware{

app: app,

keeper: k,

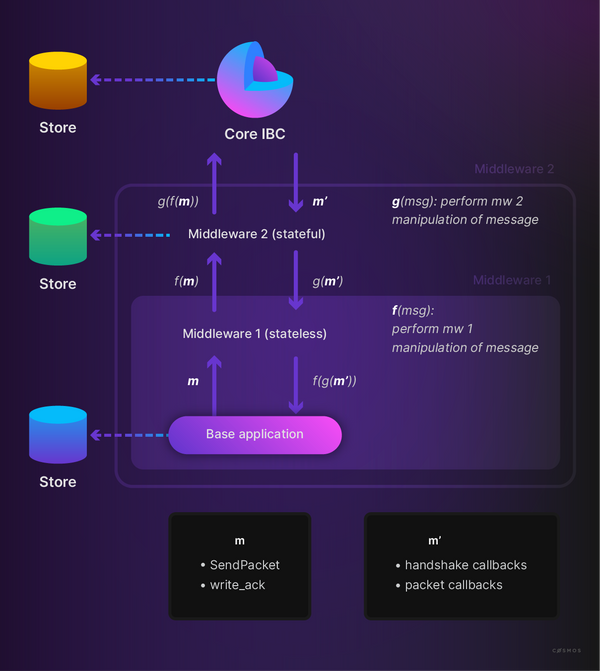
}

}

For simplification, in the snippet above, keeper.Keeper is supposed to implement the ICS4Wrapper interface. See [here (opens new window)↗](https://github.com/cosmos/ibc-go/blob/48a6ae512b4ea42c29fdf6c6f5363f50645591a2/modules/apps/29-fee/keeper/relay.go#L16-L65) for an example implementation of this interface for the ICS29 Fee Middleware module.

Below you will take a closer look at how to implement the handshake callbacks and the packet callbacks, to satisfy the IBCModule interface, and also the SendPacket, WriteAcknowledgement and GetAppVersion methods, to satisfy the ICS4Wrapper interface.

As a reminder, review once more the diagram representing the information flow between core IBC and the base application with a middleware (stack) applied:



[#Copy link](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#channel-handshake-callbacks) Channel handshake callbacks

The first type of callbacks from core IBC to the application are the channel handshake callbacks. When a middleware (stack) is applied, every piece of middleware will have access to the underlying application (base application or downstream middleware). For the handshake callbacks, function calls happen from core IBC through the middleware and then to the base application. Each middleware will call the underlying application's callback but it can execute custom application logic before doing so.

Check out the code snippets below to see this in action, or use the links to see how it is implemented for fee middleware (ICS-29).



Middleware **may** choose not to call the underlying application's callback at all. However, these should generally be limited to error cases.

[#Copy link](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#middleware-version-negotiation) Middleware version negotiation

In the case where the IBC middleware expects to speak to a compatible IBC middleware on the counterparty chain, they must **use the channel handshake to negotiate the middleware version without interfering in the version negotiation of the underlying application**.

Middleware accomplishes this by formatting the version in a JSON-encoded string containing the middleware version and the application version. The application version may as well be a JSON-encoded string, possibly including further middleware and app versions if the application stack consists of multiple middlewares wrapping a base application. The format of the version is specified in ICS-30 as follows:



Copy

{

"<middleware\_version\_key>": "<middleware\_version\_value>",

"app\_version": "<application\_version\_value>"

// ... other custom parameter fields

}

The <middleware\_version\_key> key in the JSON struct should be replaced by the actual name of the key for the corresponding middleware (e.g. fee\_version).

During the handshake callbacks, the middleware can unmarshal the version string and retrieve the middleware and application versions. It can do its negotiation logic on <middleware\_version\_value>, and pass the <application\_version\_value> to the underlying application.



Middleware that does not need to negotiate with a counterparty middleware on the remote stack will not implement the version unmarshalling and negotiation, and may simply perform its own custom logic on the callbacks without relying on the counterparty behaving similarly.

[#Copy link](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#capabilities) Capabilities

The middleware should simply pass the capability in the callback arguments along to the underlying application so that it may be claimed by the base application. The base application will then pass the capability up the stack in order to authenticate an outgoing packet/acknowledgement.

[#Copy link](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#code-snippets) Code snippets

Note that the code snippets below contain *pseudo code*, like doCustomLogic(args). Every code snippet is accompanied by a reference to the respective function in the [fee middleware IBC application (opens new window)↗](https://github.com/cosmos/ibc-go/tree/main/modules/apps/29-fee) from the official ibc-go repository.

****

**`OnChanOpenInit`**

Copy

func (im IBCMiddleware) OnChanOpenInit(

ctx sdk.Context,

order channeltypes.Order,

connectionHops []string,

portID string,

channelID string,

channelCap \*capabilitytypes.Capability,

counterparty channeltypes.Counterparty,

version string,

) (string, error) {

if version != "" {

// try to unmarshal JSON-encoded version string and pass

// the app-specific version to app callback.

// otherwise, pass version directly to app callback.

metadata, err := Unmarshal(version)

if err != nil {

// Since it is valid for the fee version to not be specified,

// the above middleware version may be for another middleware.

// Pass the entire version string onto the underlying application.

return im.app.OnChanOpenInit(

ctx,

order,

connectionHops,

portID,

channelID,

channelCap,

counterparty,

version,

)

}

else {

metadata = {

// set middleware version to default value

MiddlewareVersion: defaultMiddlewareVersion,

// allow the application to return its default version

AppVersion: "",

}

}

}

doCustomLogic()

// if the version string is empty, OnChanOpenInit is expected to return

// a default version string representing the version(s) it supports

appVersion, err := im.app.OnChanOpenInit(

ctx,

order,

connectionHops,

portID,

channelID,

channelCap,

counterparty,

metadata.AppVersion, // note you only pass app version here

)

if err != nil {

return "", err

}

version := constructVersion(metadata.MiddlewareVersion, appVersion)

return version, nil

}

See [here (opens new window)↗](https://github.com/cosmos/ibc-go/blob/48a6ae512b4ea42c29fdf6c6f5363f50645591a2/modules/apps/29-fee/ibc_middleware.go#L34-L82) for an example implementation of this callback for the ICS29 Fee Middleware module.

****

**`OnChanOpenTry`**

Copy

func (im IBCMiddleware) OnChanOpenTry(

ctx sdk.Context,

order channeltypes.Order,

connectionHops []string,

portID,

channelID string,

channelCap \*capabilitytypes.Capability,

counterparty channeltypes.Counterparty,

counterpartyVersion string,

) (string, error) {

// try to unmarshal JSON-encoded version string and pass

// the app-specific version to app callback.

// otherwise, pass version directly to app callback.

cpMetadata, err := Unmarshal(counterpartyVersion)

if err != nil {

return im.app.OnChanOpenTry(

ctx,

order,

connectionHops,

portID,

channelID,

channelCap,

counterparty,

counterpartyVersion,

)

}

doCustomLogic()

// Call the underlying application's OnChanOpenTry callback.

// The try callback must select the final app-specific version string and return it.

appVersion, err := im.app.OnChanOpenTry(

ctx,

order,

connectionHops,

portID,

channelID,

channelCap,

counterparty,

cpMetadata.AppVersion, // note you only pass counterparty app version here

)

if err != nil {

return "", err

}

// negotiate final middleware version

middlewareVersion := negotiateMiddlewareVersion(cpMetadata.MiddlewareVersion)

version := constructVersion(middlewareVersion, appVersion)

return version, nil

}

See [here (opens new window)↗](https://github.com/cosmos/ibc-go/blob/48a6ae512b4ea42c29fdf6c6f5363f50645591a2/modules/apps/29-fee/ibc_middleware.go#L84-L124) for an example implementation of this callback for the ICS29 Fee Middleware module.

****

**`OnChanOpenAck`**

Copy

func (im IBCMiddleware) OnChanOpenAck(

ctx sdk.Context,

portID,

channelID string,

counterpartyChannelID string,

counterpartyVersion string,

) error {

// try to unmarshal JSON-encoded version string and pass

// the app-specific version to app callback.

// otherwise, pass version directly to app callback.

cpMetadata, err = UnmarshalJSON(counterpartyVersion)

if err != nil {

return im.app.OnChanOpenAck(ctx, portID, channelID, counterpartyChannelID, counterpartyVersion)

}

if !isCompatible(cpMetadata.MiddlewareVersion) {

return error

}

doCustomLogic()

// call the underlying application's OnChanOpenTry callback

return im.app.OnChanOpenAck(ctx, portID, channelID, counterpartyChannelID, cpMetadata.AppVersion)

}

See [here (opens new window)↗](https://github.com/cosmos/ibc-go/blob/48a6ae512b4ea42c29fdf6c6f5363f50645591a2/modules/apps/29-fee/ibc_middleware.go#L126-L152) for an example implementation of this callback for the ICS29 Fee Middleware module.

****

**`OnChanOpenConfirm`**

Copy

func (im IBCMiddleware) OnChanOpenConfirm(

ctx sdk.Context,

portID,

channelID string,

) error {

doCustomLogic()

return im.app.OnChanOpenConfirm(ctx, portID, channelID)

}

See [here (opens new window)↗](https://github.com/cosmos/ibc-go/blob/48a6ae512b4ea42c29fdf6c6f5363f50645591a2/modules/apps/29-fee/ibc_middleware.go#L154-L162) for an example implementation of this callback for the ICS29 Fee Middleware module.

Similarly, for the channel closing:

****

**`OnChanCloseInit`**

Copy

func (im IBCMiddleware) OnChanCloseInit(

ctx sdk.Context,

portID,

channelID string,

) error {

doCustomLogic()

return im.app.OnChanCloseInit(ctx, portID, channelID)

}

See [here (opens new window)↗](https://github.com/cosmos/ibc-go/blob/48a6ae512b4ea42c29fdf6c6f5363f50645591a2/modules/apps/29-fee/ibc_middleware.go#L164-L187) for an example implementation of this callback for the ICS29 Fee Middleware module.

****

**`OnChanCloseConfirm`**

Copy

func (im IBCMiddleware) OnChanCloseConfirm(

ctx sdk.Context,

portID,

channelID string,

) error {

doCustomLogic()

return im.app.OnChanCloseConfirm(ctx, portID, channelID)

}

See [here (opens new window)↗](https://github.com/cosmos/ibc-go/blob/48a6ae512b4ea42c29fdf6c6f5363f50645591a2/modules/apps/29-fee/ibc_middleware.go#L189-L212) for an example implementation of this callback for the ICS29 Fee Middleware module.

[#Copy link](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#packet-callbacks) Packet callbacks

The middleware's packet callbacks wrap the application's packet callbacks, just like the middleware's handshake callbacks wrapped the application's handshake callbacks.



The packet callbacks are where the middleware performs most of its custom logic. The middleware may read the packet flow data and perform some additional packet handling, or it may modify the incoming data before it reaches the underlying application. This enables a wide degree of use cases, as a simple base application like token transfer can be transformed for a variety of use cases by combining it with custom middleware.

[#Copy link](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#code-snippets-2) Code snippets

Note that the code snippets below contain *pseudo code*, like doCustomLogic(args). Every code snippet is accompanied by a reference to the respective function in the [fee middleware IBC application (opens new window)↗](https://github.com/cosmos/ibc-go/tree/main/modules/apps/29-fee) from the official ibc-go repository.

****

**`OnRecvPacket`**

Copy

func (im IBCMiddleware) OnRecvPacket(

ctx sdk.Context,

packet channeltypes.Packet,

relayer sdk.AccAddress,

) ibcexported.Acknowledgement {

doCustomLogicOnPacket(packet)

ack := im.app.OnRecvPacket(ctx, packet, relayer)

doCustomLogicOnAck(ack) // middleware may modify outgoing ack

return ack

}

See [here (opens new window)↗](https://github.com/cosmos/ibc-go/blob/48a6ae512b4ea42c29fdf6c6f5363f50645591a2/modules/apps/29-fee/ibc_middleware.go#L214-L237) for an example implementation of this callback for the ICS29 Fee Middleware module.

****

**`OnAcknowledgementPacket`**

Copy

func (im IBCMiddleware) OnAcknowledgementPacket(

ctx sdk.Context,

packet channeltypes.Packet,

acknowledgement []byte,

relayer sdk.AccAddress,

) error {

doCustomLogic(packet, ack)

return im.app.OnAcknowledgementPacket(ctx, packet, ack, relayer)

}

See [here (opens new window)↗](https://github.com/cosmos/ibc-go/blob/48a6ae512b4ea42c29fdf6c6f5363f50645591a2/modules/apps/29-fee/ibc_middleware.go#L239-L292) for an example implementation of this callback for the ICS29 Fee Middleware module.

****

**`OnTimeoutPacket`**

Copy

func (im IBCMiddleware) OnTimeoutPacket(

ctx sdk.Context,

packet channeltypes.Packet,

relayer sdk.AccAddress,

) error {

doCustomLogic(packet)

return im.app.OnTimeoutPacket(ctx, packet, relayer)

}

See [here (opens new window)↗](https://github.com/cosmos/ibc-go/blob/48a6ae512b4ea42c29fdf6c6f5363f50645591a2/modules/apps/29-fee/ibc_middleware.go#L294-L334) for an example implementation of this callback for the ICS29 Fee Middleware module.

[#Copy link](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#ics-4-wrappers) ICS-4 wrappers

Middleware must also implement the ICS4Wrapper interface so that any communication from the application to the channelKeeper goes through the middleware first. Similar to the packet callbacks, the middleware may modify outgoing acknowledgements and packets in any way it wishes.

[#Copy link](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#capabilities-2) Capabilities

Earlier you saw that the handshake callbacks passed the capability in the callback arguments along to the underlying application so that it may be claimed by the base application. In the ICS4Wrapper methods, the base application will then pass the capability up the stack in order to authenticate an outgoing packet or acknowledgment.

If the middleware wishes to send a packet or acknowledgement without the involvement of the underlying application, it should be given access to the same scopedKeeper as the base application so that it can retrieve the capabilities by itself.

[#Copy link](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#code-snippets-3) Code snippets

Note that the code snippets below contain *pseudo code*, like doCustomLogic(args). Every code snippet is accompanied by a reference to the respective function in the [fee middleware IBC application (opens new window)↗](https://github.com/cosmos/ibc-go/tree/main/modules/apps/29-fee) from the official ibc-go repository.

****

**`SendPacket`**

Copy

func (im IBCMiddleware) SendPacket(

ctx sdk.Context,

chanCap \*capabilitytypes.Capability,

appPacket exported.PacketI,

) {

// middleware may modify packet

packet = doCustomLogic(appPacket)

return im.keeper.SendPacket(ctx, chanCap, packet)

}

See [here (opens new window)](https://github.com/cosmos/ibc-go/blob/48a6ae512b4ea42c29fdf6c6f5363f50645591a2/modules/apps/29-fee/ibc_middleware.go#L336-L343) for an example implementation of this function for the ICS29 Fee Middleware module.

****

**`WriteAcknowledgement`**

Copy

// only called for async acks

func (im IBCMiddleware) WriteAcknowledgement(

ctx sdk.Context,

chanCap \*capabilitytypes.Capability,

packet exported.PacketI,

ack exported.Acknowledgement,

) {

// middleware may modify acknowledgement

ack\_bytes = doCustomLogic(ack)

return im.keeper.WriteAcknowledgement(packet, ack\_bytes)

}

See [here (opens new window)](https://github.com/cosmos/ibc-go/blob/48a6ae512b4ea42c29fdf6c6f5363f50645591a2/modules/apps/29-fee/ibc_middleware.go#L345-L353) for an example implementation of this function for the ICS29 Fee Middleware module.

****

**`GetAppVersion`**

Copy

// middleware must return the underlying application version

func (im IBCMiddleware) GetAppVersion(

ctx sdk.Context,

portID,

channelID string,

) (string, bool) {

version, found := im.keeper.GetAppVersion(ctx, portID, channelID)

if !found {

return "", false

}

if !MiddlewareEnabled {

return version, true

}

// unwrap channel version

metadata, err := Unmarshal(version)

if err != nil {

panic(fmt.Errof("unable to unmarshal version: %w", err))

}

return metadata.AppVersion, true

}

See [here (opens new window)](https://github.com/cosmos/ibc-go/blob/48a6ae512b4ea42c29fdf6c6f5363f50645591a2/modules/apps/29-fee/ibc_middleware.go#L355-L358) for an example implementation of this function for the ICS29 Fee Middleware module.

synopsis

To summarize, this section has explored:

* How to integrate middleware (or a middleware stack) in app.go.
* The importance of middleware order in the middleware stack.
* How only the top-layer middleware of any stack should be connected to the IBC router.

previous

[](https://ida.interchain.io/academy/3-ibc/9-ibc-mw-intro.html)

**[IBC Middleware (OPTIONAL)](https://ida.interchain.io/academy/3-ibc/9-ibc-mw-intro.html)**

up next

**[Integrate IBC Middleware Into a Chain (OPTIONAL)](https://ida.interchain.io/academy/3-ibc/11-ibc-mw-integrate.html)**

[[](https://ida.interchain.io/academy/3-ibc/11-ibc-mw-integrate.html)](https://ida.interchain.io/academy/3-ibc/11-ibc-mw-integrate.html)

Rate this Page

icon smile

icon meh

icon frown

Would you like to add a message?

Submit

Thank you for your Feedback!

On this page

[Channel handshake callbacks](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#channel-handshake-callbacks)

[Middleware version negotiation](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#middleware-version-negotiation)

[Capabilities](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#capabilities)

[Code snippets](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#code-snippets)

[Packet callbacks](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#packet-callbacks)

[Code snippets](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#code-snippets-2)

[ICS-4 wrappers](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#ics-4-wrappers)

[Capabilities](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#capabilities-2)

[Code snippets](https://ida.interchain.io/academy/3-ibc/10-ibc-mw-develop.html#code-snippets-3)

#### **Get Cosmos updates**

Unsubscribe at any time. [Privacy Policy↗](https://v1.cosmos.network/privacy)

     Next

Documentation

[Cosmos SDK](https://docs.cosmos.network/)[Cosmos Hub](https://hub.cosmos.network/)[CometBFT](https://docs.cometbft.com/)[IBC Protocol](https://ibc.cosmos.network/)

Community

[Interchain blog](https://blog.cosmos.network/)[Forum](https://forum.cosmos.network/)[Discord](https://discord.gg/cosmosnetwork)

Contributing

[Source code on GitHub](https://github.com/cosmos/sdk-tutorials)

[](https://ida.interchain.io/)

[Interchain Developer Academy](https://ida.interchain.io/)

**[](https://blog.cosmos.network/)[](https://twitter.com/cosmos)[](https://discord.gg/cosmosnetwork)[](https://www.linkedin.com/company/interchain-foundation/about/)[](https://reddit.com/r/cosmosnetwork)[](https://t.me/cosmosproject)[](https://www.youtube.com/c/CosmosProject)**



Dark mode

† This website is maintained by the Interchain Foundation (ICF). The contents and opinions of this website are those of the ICF. The ICF provides links to cryptocurrency exchanges as a service to the public. The ICF does not warrant that the information provided by these websites is correct, complete, and up-to-date. The ICF is not responsible for their content and expressly rejects any liability for damages of any kind resulting from the use, reference to, or reliance on any information contained within these websites.

Cosmos is a registered trademark of the [Interchain Foundation.](https://interchain.io/)[Privacy](https://v1.cosmos.network/privacy)