By using this website, you agree to our [Cookie Policy](https://www.cookiesandyou.com/).



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

[Interchain Developer Academy](https://ida.interchain.io/)/[Interchain Developer Academy](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)



Search

[Interchain Developer Academy](https://ida.interchain.io/)[Interchain Developer Academy](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

Search



Filters

Interchain Developer Academy

[](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Week 0 - Getting Started](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Getting Started](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Blockchain 101](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Blockchain History](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Public and Managed Blockchains](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Consensus in Distributed Networks](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Cryptography](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Self-Assessment Quiz](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Go Introduction - First Steps](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Go Basics](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Go Interfaces](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Control Structures in Go](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Arrays and Slices in Go](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Standard Packages in Go](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Concurrency in Go](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Good-To-Know Dev Terms](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Docker Introduction](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Week 1 - Introduction to the Interchain](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Introduction to the Interchain](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Blockchain Technology and the Interchain](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[The Interchain Ecosystem](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Getting ATOM and Staking It](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[A Blockchain App Architecture](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Accounts](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Transactions](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Messages](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Modules](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Protobuf](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Multistore and Keepers](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[BaseApp](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Queries](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Events](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Context](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Testing](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Relaying with IBC](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Interchain Security](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Bridges](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Migrations](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Week 1 Quiz](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Week 2 - First Steps](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[First Steps](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Setup Your Work Environment](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Run a Node, API, and CLI](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Ignite CLI](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Exercise - Make a Checkers Blockchain](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Store Object](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Create Custom Messages](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Create and Save a Game Properly](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Add a Way to Make a Move](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Emit Game Information](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Record the Game Winner](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Week 2 Exercise](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Week 3 - Introduction to IBC and CosmJS](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Introduction to IBC and CosmJS](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[What is IBC?](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[IBC/TAO - Connections (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[IBC/TAO - Channels (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[IBC/TAO - Clients (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[IBC Token Transfer](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Interchain Accounts (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[IBC Middleware (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Create a Custom IBC Middleware (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Integrate IBC Middleware Into a Chain (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[IBC Tooling](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[What is CosmJS?](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Your First CosmJS Actions](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Compose Complex Transactions](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Learn to Integrate Keplr](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Create Custom CosmJS Interfaces](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Week 4 - Ignite CLI and IBC Advanced](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Ignite CLI and IBC Advanced](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Keep an Up-To-Date Game Deadline](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Keep Track Of How Many Moves Have Been Played](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Put Your Games in Order](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Auto-Expiring Games](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Let Players Set a Wager](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Handle wager payments](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Integration tests](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Incentivize Players](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Help Find a Correct Move](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Play With Cross-Chain Tokens](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Understand IBC Denoms](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Go Relayer](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Hermes Relayer](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Week 5 - CosmJS Advanced](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[CosmJS Advanced](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Create Custom Objects](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Create Custom Messages](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Get an External GUI](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Integrate CosmJS and Keplr](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Backend Script for Game Indexing](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Week 6 - IBC Deep Dive](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[IBC Deep Dive](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[IBC Application Developer Introduction](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Make a Module IBC-Enabled](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Adding Packet and Acknowledgment Data](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Extend the Checkers Game With a Leaderboard](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Create a Leaderboard Chain](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Week 7 - From Code to MVP to Production and Migrations](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[From Code to MVP to Production and Migrations](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Run in Production](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Prepare the Software to Run](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Prepare a Validator and Keys](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Prepare Where the Node Starts](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Prepare and Connect to Other Nodes](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Configure, Run, and Set Up a Service](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Prepare and Do Migrations](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Simulate Production in Docker](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Tally Player Info After Production](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Add a Leaderboard as a Module](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Migrate the Leaderboard Module After Production](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Simulate a Migration in Docker](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Final Exam](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[What's Next?](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

[Continue Your Interchain Journey](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html)

Docs Version Switcher

On this page

[What you will be building and why](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html#what-you-will-be-building-and-why)

[Adding a local leaderboard module to the checkers chain](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html#adding-a-local-leaderboard-module-to-the-checkers-chain)

[Unit tests](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html#unit-tests)

[Forwarding player information via IBC](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html#forwarding-player-information-via-ibc)

[#Copy link](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html#extend-the-checkers-game-with-a-leaderboard) **Extend the Checkers Game With a Leaderboard**



In this section, you will learn:

* How to make an existing chain IBC-enabled.
* How to extend your chains with additional modules.

[#Copy link](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html#what-you-will-be-building-and-why) What you will be building and why

The checkers blockchain you have built has the ability to create games, play them, forfeit them, and wager on them (potentially with cross-chain tokens). A further optimization would be to include a leaderboard. This could be executed locally on the checkers blockchain to rank the best players on the checkers blockchain. You can see an example of this in the [migration sections](https://ida.interchain.io/hands-on-exercise/4-run-in-prod/2-migration-info.html).

But what if there is more than one checkers chain? Or better yet, other game chains that allow players to play competitive games. Would it not be great to enable a standard to send the game data from the local game chain to an application-specific chain that keeps a global leaderboard? This is exactly what you will be building in the next few sections.



Remember the [appchain thesis that is an integral part of the Interchain philosophy](https://ida.interchain.io/academy/3-ibc/1-what-is-ibc.html#internet-of-blockchains) - where every application has its own chain and can be optimized for the application-specific logic it executes. Then IBC can be used to interoperate between all the chains that have specialized functionality. This is the idea behind the prototype checkers and leaderboard chains you're building, enabling IBC packets to be sent between those chains to create cross-chain applications.

[#Copy link](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html#adding-a-local-leaderboard-module-to-the-checkers-chain) Adding a local leaderboard module to the checkers chain

Currently, your checkers game contains the checkers module but is not IBC-enabled. It is now time to extend your checkers game with a leaderboard by adding a new module to make it IBC-enabled.

Let’s dive right into it.

Go to your checkers folder and make sure that you are checked out on the [cosmjs-elements (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/tree/cosmjs-elements) branch.

In the checkers chain folder, you can scaffold a leaderboard module with Ignite:

**Local**

**Docker**



Copy

$ ignite scaffold module leaderboard --ibc

Copy

$ docker run --rm -it \

-v $(pwd):/checkers \

-w /checkers \

ignitehq/cli:0.22.1 \

scaffold module leaderboard --ibc

In order to create and maintain a leaderboard, you need to store the player information. Scaffold a structure with:

**Local**

**Docker**



Copy

$ ignite scaffold map playerInfo \

wonCount:uint lostCount:uint forfeitedCount:uint \

dateUpdated:string \

--module leaderboard \

--no-message

Copy

$ docker run --rm -it \

-v $(pwd):/leaderboard \

-w /leaderboard \

ignitehq/cli:0.22.1 \

scaffold map playerInfo \

wonCount:uint lostCount:uint forfeitedCount:uint \

dateUpdated:string \

--module leaderboard \

--no-message

Now you can use this structure to create the board itself:

**Local**

**Docker**



Copy

$ ignite scaffold single board \

PlayerInfo:PlayerInfo \

--module leaderboard \

--no-message

Copy

$ docker run --rm -it \

-v $(pwd):/leaderboard \

-w /leaderboard \

ignitehq/cli:0.22.1 \

scaffold single board \

PlayerInfo:PlayerInfo \

--module leaderboard \

--no-message

The structures created by Ignite are [nullable types (opens new window)↗](https://en.wikipedia.org/wiki/Nullable_type) by default, but you do not want that. So a few adjustments are needed - especially because you do not have a null value for an address.

You need to make the adjustments in the Protobuf files proto/leaderboard/board.proto and proto/leaderboard/genesis.proto. Make sure to import gogoproto/gogo.proto and use [(gogoproto.nullable) = false]; for the PlayerInfo and the [Board (opens new window)↗](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/proto/leaderboard/genesis.proto#L17).

For example, for proto/leaderboard/board.proto try this:



Copy

syntax = "proto3";

package b9lab.checkers.leaderboard;

option go\_package = "github.com/b9lab/checkers/x/leaderboard/types";

import "leaderboard/player\_info.proto";

+ import "gogoproto/gogo.proto";

message Board {

- PlayerInfo playerInfo = 1;

+ repeated PlayerInfo playerInfo = 1 [(gogoproto.nullable) = false];

}

modular /

b9-checkers-academy-draft /

... /

leaderboard /

board.proto

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/proto/leaderboard/board.proto" \l "L9" \t "_blank)



After re-compilation, you will also have to modify the x/leaderboard/genesis.go. In it, look for:



Copy

- if genState.Board != nil {

- k.SetBoard(ctx, \*genState.Board)

- }

+ k.SetBoard(ctx, genState.Board)

modular /

b9-checkers-academy-draft /

... /

leaderboard /

genesis.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/leaderboard/genesis.go" \l "L17" \t "_blank)

And:



Copy

if found {

- genesis.Board = &board

+ genesis.Board = board

}

modular /

b9-checkers-academy-draft /

... /

leaderboard /

genesis.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/leaderboard/genesis.go" \l "L44" \t "_blank)

Next, in the x/leaderboard/genesis\_test.go, look for:



Copy

- Board: &types.Board{

- PlayerInfo: new(types.PlayerInfo),

+ Board: types.Board{

+ PlayerInfo: []types.PlayerInfo{},

},

modular /

b9-checkers-academy-draft /

... /

leaderboard /

genesis\_test.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/leaderboard/genesis_test.go" \l "L25-L27" \t "_blank)

Next, in x/leaderboard/types/genesis.go, look for:



Copy

- Board: nil,

+ Board: Board{

+ PlayerInfo: []PlayerInfo{},

+ },

modular /

b9-checkers-academy-draft /

... /

types /

genesis.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/leaderboard/types/genesis.go" \l "L16-L18" \t "_blank)

Now, in its associated test:



Copy

- Board: &types.Board{

- PlayerInfo: new(types.PlayerInfo),

- },

+ Board: types.Board{

+ PlayerInfo: []types.PlayerInfo{},

+ },

modular /

b9-checkers-academy-draft /

... /

types /

genesis\_test.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/leaderboard/types/genesis_test.go" \l "L33-L35" \t "_blank)

Lastly in:



Copy

- board := &types.Board{}

+ board := types.Board{}

...

- return network.New(t, cfg), \*state.Board

+ return network.New(t, cfg), state.Board

modular /

b9-checkers-academy-draft /

... /

cli /

query\_board\_test.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/leaderboard/client/cli/query_board_test.go" \l "L24-L30" \t "_blank)

Continue preparing your new leaderboard module.

The checkers module is the authority when it comes to who won and who lost; the leaderboard module is the authority when it comes to how to tally scores and rank players. Therefore, the leaderboard module will only expose to the checkers module functions to inform on wins and losses, like MustAddWonGameResultToPlayer(...).

To achieve this, first you need to write those functions. Create a x/leaderboard/keeper/player\_info\_handler.go file with the following code:



Copy

package keeper

import (

"github.com/b9lab/checkers/x/leaderboard/types"

sdk "github.com/cosmos/cosmos-sdk/types"

)

func mustAddDeltaGameResultToPlayer(

k \*Keeper,

ctx sdk.Context,

player sdk.AccAddress,

wonDelta uint64,

lostDelta uint64,

forfeitedDelta uint64,

) (playerInfo types.PlayerInfo) {

playerInfo, found := k.GetPlayerInfo(ctx, player.String())

if !found {

playerInfo = types.PlayerInfo{

Index: player.String(),

WonCount: 0,

LostCount: 0,

ForfeitedCount: 0,

DateUpdated: ctx.BlockTime().UTC().Format(types.TimeLayout),

}

}

playerInfo.WonCount += wonDelta

playerInfo.LostCount += lostDelta

playerInfo.ForfeitedCount += forfeitedDelta

k.SetPlayerInfo(ctx, playerInfo)

return playerInfo

}

func (k Keeper) MustAddWonGameResultToPlayer(ctx sdk.Context, player sdk.AccAddress) types.PlayerInfo {

return mustAddDeltaGameResultToPlayer(&k, ctx, player, 1, 0, 0)

}

func (k Keeper) MustAddLostGameResultToPlayer(ctx sdk.Context, player sdk.AccAddress) types.PlayerInfo {

return mustAddDeltaGameResultToPlayer(&k, ctx, player, 0, 1, 0)

}

func (k Keeper) MustAddForfeitedGameResultToPlayer(ctx sdk.Context, player sdk.AccAddress) types.PlayerInfo {

return mustAddDeltaGameResultToPlayer(&k, ctx, player, 0, 0, 1)

}

modular /

b9-checkers-academy-draft /

... /

keeper /

player\_info\_handler.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/leaderboard/keeper/player_info_handler.go" \t "_blank)

For the code above to function, you need to define TimeLayout in x/leaderboard/types/keys.go. Add the following piece of code at the end of the file:



Copy

const (

TimeLayout = "2006-01-02 15:04:05.999999999 +0000 UTC"

LeaderboardWinnerLength = uint64(100)

)

modular /

b9-checkers-academy-draft /

... /

types /

keys.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/leaderboard/types/keys.go" \l "L39-L42" \t "_blank)

Now it is time to allow the checkers module access to the leaderboard module. This is very similar to what you did when giving access to the bank keeper when handling wager tokens.

Declare the leaderboard functions that the checkers needs:



Copy

import (

leaderboardTypes "github.com/b9lab/checkers/x/leaderboard/types"

)

type CheckersLeaderboardKeeper interface {

MustAddWonGameResultToPlayer(ctx sdk.Context, player sdk.AccAddress) leaderboardTypes.PlayerInfo

MustAddLostGameResultToPlayer(ctx sdk.Context, player sdk.AccAddress) leaderboardTypes.PlayerInfo

MustAddForfeitedGameResultToPlayer(ctx sdk.Context, player sdk.AccAddress) leaderboardTypes.PlayerInfo

}

modular /

b9-checkers-academy-draft /

... /

types /

expected\_keepers.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/checkers/types/expected_keepers.go" \l "L26-L30" \t "_blank)

Add this keeper interface to checkers, modify x/checkers/keeper/keeper.go, and include the leaderboard keeper:



Copy

type (

Keeper struct {

bank types.BankEscrowKeeper

+ board types.CheckersLeaderboardKeeper

cdc codec.BinaryCodec

storeKey sdk.StoreKey

memKey sdk.StoreKey

paramstore paramtypes.Subspace

}

)

...

func NewKeeper(

bank types.BankEscrowKeeper,

+ board types.CheckersLeaderboardKeeper,

cdc codec.BinaryCodec,

storeKey,

memKey sdk.StoreKey,

ps paramtypes.Subspace,

) \*Keeper {

// set KeyTable if it has not already been set

if !ps.HasKeyTable() {

ps = ps.WithKeyTable(types.ParamKeyTable())

}

return &Keeper{

bank: bank,

+ board: board,

cdc: cdc,

storeKey: storeKey,

memKey: memKey,

paramstore: ps,

}

}

modular /

b9-checkers-academy-draft /

... /

keeper /

keeper.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/checkers/keeper/keeper.go" \l "L19" \t "_blank)

Make sure the app builds it correctly. Look for app.CheckersKeeper in app/app.go and modify it to include app.LeaderboardKeeper:



Copy

app.CheckersKeeper = \*checkersmodulekeeper.NewKeeper(

app.BankKeeper,

+ app.LeaderboardKeeper,

appCodec,

...

)

modular /

b9-checkers-academy-draft /

app /

app.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/app/app.go" \l "L440" \t "_blank)

You want to store a *win* plus either a *loss* or a *forfeit* when a game ends. Therefore, you should create some helper functions in checkers that call the leaderboard module. Create a x/checkers/keeper/player\_info\_handler.go file with the following code:



Copy

package keeper

import (

"fmt"

rules "github.com/b9lab/checkers/x/checkers/rules"

"github.com/b9lab/checkers/x/checkers/types"

sdk "github.com/cosmos/cosmos-sdk/types"

)

func getWinnerAndLoserAddresses(storedGame \*types.StoredGame) (winnerAddress sdk.AccAddress, loserAddress sdk.AccAddress) {

if storedGame.Winner == rules.PieceStrings[rules.NO\_PLAYER] {

panic(types.ErrThereIsNoWinner.Error())

}

redAddress, err := storedGame.GetRedAddress()

if err != nil {

panic(err.Error())

}

blackAddress, err := storedGame.GetBlackAddress()

if err != nil {

panic(err.Error())

}

if storedGame.Winner == rules.PieceStrings[rules.RED\_PLAYER] {

winnerAddress = redAddress

loserAddress = blackAddress

} else if storedGame.Winner == rules.PieceStrings[rules.BLACK\_PLAYER] {

winnerAddress = blackAddress

loserAddress = redAddress

} else {

panic(fmt.Sprintf(types.ErrWinnerNotParseable.Error(), storedGame.Winner))

}

return winnerAddress, loserAddress

}

func (k \*Keeper) MustRegisterPlayerWin(ctx sdk.Context, storedGame \*types.StoredGame) {

winnerAddress, loserAddress := getWinnerAndLoserAddresses(storedGame)

k.board.MustAddWonGameResultToPlayer(ctx, winnerAddress)

k.board.MustAddLostGameResultToPlayer(ctx, loserAddress)

}

func (k \*Keeper) MustRegisterPlayerForfeit(ctx sdk.Context, storedGame \*types.StoredGame) {

winnerAddress, loserAddress := getWinnerAndLoserAddresses(storedGame)

k.board.MustAddWonGameResultToPlayer(ctx, winnerAddress)

k.board.MustAddForfeitedGameResultToPlayer(ctx, loserAddress)

}

modular /

b9-checkers-academy-draft /

... /

keeper /

player\_info\_handler.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/checkers/keeper/player_info_handler.go" \t "_blank)

Do not forget to add the new errors in x/checkers/types/errors.go:



Copy

ErrWinnerNotParseable = sdkerrors.Register(ModuleName, 1118, "winner is not parseable: %s")

ErrThereIsNoWinner = sdkerrors.Register(ModuleName, 1119, "there is no winner")

ErrInvalidDateAdded = sdkerrors.Register(ModuleName, 1120, "dateAdded cannot be parsed: %s")

ErrCannotAddToLeaderboard = sdkerrors.Register(ModuleName, 1121, "cannot add to leaderboard: %s")

modular /

b9-checkers-academy-draft /

... /

types /

errors.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/checkers/types/errors.go" \l "L29-L32" \t "_blank)

With the helper functions ready, you can call them where needed. Add the call for a *win* in x/checkers/keeper/msg\_server\_play\_move.go:



Copy

func (k msgServer) PlayMove(goCtx context.Context, msg \*types.MsgPlayMove) (\*types.MsgPlayMoveResponse, error) {

...

lastBoard := game.String()

if storedGame.Winner == rules.PieceStrings[rules.NO\_PLAYER] {

k.Keeper.SendToFifoTail(ctx, &storedGame, &systemInfo)

storedGame.Board = lastBoard

} else {

k.Keeper.RemoveFromFifo(ctx, &storedGame, &systemInfo)

storedGame.Board = ""

k.Keeper.MustPayWinnings(ctx, &storedGame)

+ // Here you can register a win

+ k.Keeper.MustRegisterPlayerWin(ctx, &storedGame)

}

...

modular /

b9-checkers-academy-draft /

... /

keeper /

msg\_server\_play\_move.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/checkers/keeper/msg_server_play_move.go" \l "LL80" \t "_blank)

Add the call for a *forfeit* in x/checkers/keeper/end\_block\_server\_game.go:



Copy

func (k Keeper) ForfeitExpiredGames(goCtx context.Context) {

...

if deadline.Before(ctx.BlockTime()) {

// Game is past deadline

k.RemoveFromFifo(ctx, &storedGame, &systemInfo)

lastBoard := storedGame.Board

if storedGame.MoveCount <= 1 {

// No point in keeping a game that was never really played

k.RemoveStoredGame(ctx, gameIndex)

if storedGame.MoveCount == 1 {

k.MustRefundWager(ctx, &storedGame)

}

} else {

storedGame.Winner, found = opponents[storedGame.Turn]

if !found {

panic(fmt.Sprintf(types.ErrCannotFindWinnerByColor.Error(), storedGame.Turn))

}

k.MustPayWinnings(ctx, &storedGame)

+ // Here you can register a forfeit

+ k.MustRegisterPlayerForfeit(ctx, &storedGame)

storedGame.Board = ""

k.SetStoredGame(ctx, storedGame)

}

...

modular /

b9-checkers-academy-draft /

... /

keeper /

end\_block\_server\_game.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/checkers/keeper/end_block_server_game.go" \l "L57" \t "_blank)

That will get the job done, and add the player's *win*, *loss*, or *forfeit* counts to the store.

If you did the [migration part](https://ida.interchain.io/hands-on-exercise/4-run-in-prod/2-migration-info.html) of this hands-on exercise, you may notice that, here, although the player info is updated, the leaderboard is not. This is deliberate in order to show a different workflow.

Here, the leaderboard is updated on-demand by adding the signers of a message as candidates to the leaderboard. Scaffold a new message:

**Local**

**Docker**



Copy

$ ignite scaffold message updateBoard --module leaderboard

Copy

$ docker run --rm -it \

-v $(pwd):/leaderboard \

-w /leaderboard \

ignitehq/cli:0.22.1 \

scaffold message updateBoard --module leaderboard

Again, you can first create some helper functions in x/leaderboard/types/board.go:



Copy

func ParseDateAddedAsTime(dateAdded string) (dateAddedParsed time.Time, err error) {

dateAddedParsed, errDateAdded := time.Parse(types.TimeLayout, dateAdded)

return dateAddedParsed, sdkerrors.Wrapf(errDateAdded, types.ErrInvalidDateAdded.Error(), dateAdded)

}

func SortPlayerInfo(playerInfoList []types.PlayerInfo) {

sort.SliceStable(playerInfoList[:], func(i, j int) bool {

if playerInfoList[i].WonCount > playerInfoList[j].WonCount {

return true

}

if playerInfoList[i].WonCount < playerInfoList[j].WonCount {

return false

}

firstPlayerTime, \_ := ParseDateAddedAsTime(playerInfoList[i].DateUpdated)

secondPlayerTime,\_ := ParseDateAddedAsTime(playerInfoList[j].DateUpdated)

return firstPlayerTime.After(secondPlayerTime)

})

}

func UpdatePlayerInfoList(winners []PlayerInfo, candidates []PlayerInfo) (updated []PlayerInfo) {

found := false

for \_, candidate := range candidates {

for winnerIndex, winner := range winners {

if winner.Index == candidate.Index {

winners[winnerIndex] = candidate

found = true

break

}

}

if !found {

updated = append(winners, candidate)

} else {

updated = winners

}

}

SortPlayerInfo(updated)

if LeaderboardWinnerLength < uint64(len(updated)) {

updated = updated[:LeaderboardWinnerLength]

}

return updated

}

modular /

b9-checkers-academy-draft /

... /

types /

board.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/leaderboard/types/board.go" \l "L40-L70" \t "_blank)

The function that sorts players is rather inefficient, as it parses dates a lot. To optimize this part, you would have to introduce a new type with the date already parsed. See the [migration section](https://ida.interchain.io/hands-on-exercise/4-run-in-prod/2-migration-info.html) for an example.

If it cannot parse the date information, it will return an error that you need to declare in x/leaderboard/types/errors.go:



Copy

ErrInvalidDateAdded = sdkerrors.Register(ModuleName, 1120, "dateAdded cannot be parsed: %s")

modular /

b9-checkers-academy-draft /

... /

types /

errors.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/leaderboard/types/errors.go" \l "L14" \t "_blank)

Now you need to call what you created in x/leaderboard/keeper/msg\_server\_update\_board.go:



Copy

package keeper

import (

"context"

"github.com/b9lab/checkers/x/leaderboard/types"

sdk "github.com/cosmos/cosmos-sdk/types"

)

func (k msgServer) UpdateBoard(goCtx context.Context, msg \*types.MsgUpdateBoard) (\*types.MsgUpdateBoardResponse, error) {

ctx := sdk.UnwrapSDKContext(goCtx)

- // TODO: Handling the message

- \_ = ctx

+ board, found := k.GetBoard(ctx)

+ if !found {

+ panic(types.ErrBoardNotFound)

+ }

+ playerInfoList := board.PlayerInfo

+ candidates := make([]types.PlayerInfo, 0, len(msg.GetSigners()))

+ for \_, signer := range msg.GetSigners() {

+ candidate, found := k.GetPlayerInfo(ctx, signer.String())

+ if found {

+ candidates = append(candidates, candidate)

+ }

+ }

+ if len(candidates) == 0 {

+ return nil, types.ErrCandidateNotFound

+ }

+ updated := types.UpdatePlayerInfoList(playerInfoList, candidates)

+ board.PlayerInfo = updated

+ k.SetBoard(ctx, board)

return &types.MsgUpdateBoardResponse{}, nil

}

modular /

b9-checkers-academy-draft /

... /

keeper /

msg\_server\_update\_board.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/leaderboard/keeper/msg_server_update_board.go" \l "L13-L14" \t "_blank)

Also call the two new errors:



Copy

ErrBoardNotFound = sdkerrors.Register(ModuleName, 1502, "board not found")

ErrCandidateNotFound = sdkerrors.Register(ModuleName, 1503, "candidate not found")

modular /

b9-checkers-academy-draft /

... /

types /

errors.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/leaderboard/types/errors.go" \l "L15-L16" \t "_blank)

That is it! Now the checkers blockchain can keep track of player information, and create or update the leaderboard based on player input.

[#Copy link](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html#unit-tests) Unit tests

You have created a new expected keeper. Have Mockgen create its mocks by reusing the make command prepared earlier:

**Local**

**Docker**



Copy

$ make mock-expected-keepers

Copy

$ docker run --rm -it \

-v $(pwd):/checkers \

-w /checkers \

checkers\_i \

make mock-expected-keepers

Update the checkers keeper factory for tests:



Copy

func CheckersKeeper(t testing.TB) (\*keeper.Keeper, sdk.Context) {

- return CheckersKeeperWithMocks(t, nil)

+ return CheckersKeeperWithMocks(t, nil, nil)

}

- func CheckersKeeperWithMocks(t testing.TB, bank \*mock\_types.MockBankEscrowKeeper) (\*keeper.Keeper, sdk.Context) {

+ func CheckersKeeperWithMocks(t testing.TB, bank \*mock\_types.MockBankEscrowKeeper, leaderboard \*mock\_types.MockCheckersLeaderboardKeeper) (\*keeper.Keeper, sdk.Context) {

...

k := keeper.NewKeeper(

bank,

+ leaderboard,

cdc,

storeKey,

memStoreKey,

paramsSubspace,

)

...

return k, ctx

}

modular /

b9-checkers-academy-draft /

... /

keeper /

checkers.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/testutil/keeper/checkers.go" \l "L21-L60" \t "_blank)

Because of the change of signature of this function, you need to adjust wherever it is called, like so:



Copy

func setupMsgServerWithOneGameForPlayMove(t testing.TB) (types.MsgServer, keeper.Keeper, context.Context,

- \*gomock.Controller, \*mock\_types.MockBankEscrowKeeper) {

+ \*gomock.Controller, \*mock\_types.MockBankEscrowKeeper, \*mock\_types.MockCheckersLeaderboardKeeper) {

ctrl := gomock.NewController(t)

bankMock := mock\_types.NewMockBankEscrowKeeper(ctrl)

- k, ctx := keepertest.CheckersKeeperWithMocks(t, bankMock)

+ leaderboardMock := mock\_types.NewMockCheckersLeaderboardKeeper(ctrl)

+ k, ctx := keepertest.CheckersKeeperWithMocks(t, bankMock, leaderboardMock)

checkers.InitGenesis(ctx, \*k, \*types.DefaultGenesis())

...

- return server, \*k, context, ctrl, bankMock

+ return server, \*k, context, ctrl, bankMock, leaderboardMock

}

modular /

b9-checkers-academy-draft /

... /

keeper /

msg\_server\_play\_move\_test.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/checkers/keeper/msg_server_play_move_test.go" \l "L17-L34" \t "_blank)

You must also change where it is used. Mostly like this, when the leaderboard is not called:



Copy

func TestPlayMove(t \*testing.T) {

- msgServer, \_, context, ctrl, escrow := setupMsgServerWithOneGameForPlayMove(t)

+ msgServer, \_, context, ctrl, escrow, \_ := setupMsgServerWithOneGameForPlayMove(t)

defer ctrl.Finish()

...

}

modular /

b9-checkers-academy-draft /

... /

keeper /

msg\_server\_play\_move\_test.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/checkers/keeper/msg_server_play_move_test.go" \l "L37" \t "_blank)

Like this, when the leaderboard is called but you are not looking to confirm the calls:



Copy

func TestPlayMoveUpToWinner(t \*testing.T) {

- msgServer, keeper, context, ctrl, escrow := setupMsgServerWithOneGameForPlayMove(t)

+ msgServer, keeper, context, ctrl, escrow, board := setupMsgServerWithOneGameForPlayMove(t)

ctx := sdk.UnwrapSDKContext(context)

defer ctrl.Finish()

escrow.ExpectAny(context)

+ board.ExpectAny(context)

...

}

modular /

b9-checkers-academy-draft /

... /

keeper /

msg\_server\_play\_move\_winner\_test.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/checkers/keeper/msg_server_play_move_winner_test.go" \l "L88-L92" \t "_blank)

This introduces a new function, so (as you did for the bank keeper mock) you add the missing helpers to define expectations. For instance:



Copy

func (escrow \*MockCheckersLeaderboardKeeper) ExpectAny(context context.Context) {

escrow.EXPECT().MustAddWonGameResultToPlayer(sdk.UnwrapSDKContext(context), gomock.Any()).AnyTimes()

escrow.EXPECT().MustAddLostGameResultToPlayer(sdk.UnwrapSDKContext(context), gomock.Any()).AnyTimes()

escrow.EXPECT().MustAddForfeitedGameResultToPlayer(sdk.UnwrapSDKContext(context), gomock.Any()).AnyTimes()

}

func (escrow \*MockCheckersLeaderboardKeeper) ExpectWin(context context.Context, who string) \*gomock.Call {

whoAddr, err := sdk.AccAddressFromBech32(who)

if err != nil {

panic(err)

}

return escrow.EXPECT().MustAddWonGameResultToPlayer(sdk.UnwrapSDKContext(context), whoAddr)

}

...

modular /

b9-checkers-academy-draft /

... /

mock\_types /

leaderboard\_helpers.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/testutil/mock_types/leaderboard_helpers.go" \t "_blank)

Do not forget to add tests to check that the leaderboard is called as expected:



Copy

func TestPlayMoveUpToWinnerCalledLeaderboard(t \*testing.T) {

msgServer, \_, context, ctrl, escrow, board := setupMsgServerWithOneGameForPlayMove(t)

defer ctrl.Finish()

escrow.ExpectAny(context)

board.ExpectWin(context, bob).Times(1)

board.ExpectLoss(context, carol).Times(1)

playAllMoves(t, msgServer, context, "1", game1Moves)

}

modular /

b9-checkers-academy-draft /

... /

keeper /

msg\_server\_play\_move\_winner\_test.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/checkers/keeper/msg_server_play_move_winner_test.go" \l "L145-L153" \t "_blank)

Do not forget to add some [unit tests (opens new window)↗](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/leaderboard/keeper/msg_server_update_board_test.go) for your leaderboard keeper too.

[#Copy link](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html#forwarding-player-information-via-ibc) Forwarding player information via IBC

It is time to look at how you can forward the player information via IBC.



Remember, you created the module with the --ibc flag.

You can scaffold an IBC transaction with:

**Local**

**Docker**



Copy

$ ignite scaffold packet candidate \

PlayerInfo:PlayerInfo \

--module leaderboard

Copy

$ docker run --rm -it \

-v $(pwd):/leaderboard \

-w /leaderboard \

ignitehq/cli:0.22.1 \

scaffold packet candidate \

PlayerInfo:PlayerInfo \

--module leaderboard

How the message constructor was created makes the player information a parameter. However, you do not want arbitrary player information, but instead want to fetch the creator's player information from the store. To do this, make a small adjustment to x/leaderboard/client/cli/tx\_candidate.go. Look for the following lines and remove them:



Copy

func CmdSendCandidate() \*cobra.Command {

cmd := &cobra.Command{

- Use: "send-candidate [src-port] [src-channel] [player-info]",

+ Use: "send-candidate [src-port] [src-channel]",

Short: "Send a candidate over IBC",

- Args: cobra.ExactArgs(3),

+ Args: cobra.ExactArgs(2),

RunE: func(cmd \*cobra.Command, args []string) error {

...

- argPlayerInfo := new(types.PlayerInfo)

- err = json.Unmarshal([]byte(args[2]), argPlayerInfo)

- if err != nil {

- return err

- }

...

- msg := types.NewMsgSendCandidate(creator, srcPort, srcChannel, argPlayerInfo, timeoutTimestamp)

+ msg := types.NewMsgSendCandidate(creator, srcPort, srcChannel, timeoutTimestamp)

...

},

}

...

return cmd

}

modular /

b9-checkers-academy-draft /

... /

cli /

tx\_candidate.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/leaderboard/client/cli/tx_candidate.go" \l "L16" \t "_blank)

You will also need to remove the import of encoding/json because it is not used anymore, and you should remove the parameter argPlayerInfo from the types.NewMsgSendCandidate(...) call, from [function (opens new window)↗](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/leaderboard/types/messages_candidate.go#L12-L22), and not least from MsgSendCandidate itself:



Copy

message MsgSendCandidate {

...

uint64 timeoutTimestamp = 4;

- PlayerInfo playerInfo = 5;

}

modular /

b9-checkers-academy-draft /

... /

leaderboard /

tx.proto

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/proto/leaderboard/tx.proto" \l "L23-L28" \t "_blank)

The last step is to implement the logic to fetch and send the player information in x/leaderboard/keeper/msg\_server\_candidate.go:



Copy

func (k msgServer) SendCandidate(goCtx context.Context, msg \*types.MsgSendCandidate) (\*types.MsgSendCandidateResponse, error) {

ctx := sdk.UnwrapSDKContext(goCtx)

- // TODO: logic before transmitting the packet

+ // get the Player data

+ playerInfo, found := k.GetPlayerInfo(ctx, msg.Creator)

+ if !found {

+ return nil, types.ErrCandidateNotFound

+ }

// Construct the packet

var packet types.CandidatePacketData

- packet.PlayerInfo = msg.PlayerInfo

+ packet.PlayerInfo = &playerInfo

...

}

modular /

b9-checkers-academy-draft /

... /

keeper /

msg\_server\_candidate.go

[View source→](https://github.com/b9lab/cosmos-ibc-docker/blob/main/modular/b9-checkers-academy-draft/x/leaderboard/keeper/msg_server_candidate.go" \t "_blank)

You do not handle received packets, because this module is only meant for sending player information to a separate leaderboard chain, which you will create next.

previous

[](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/5-ibc-app-packets.html)

**[Adding Packet and Acknowledgment Data](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/5-ibc-app-packets.html)**

up next

**[Create a Leaderboard Chain](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/7-ibc-app-leaderboard.html)**

[[](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/7-ibc-app-leaderboard.html)](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/7-ibc-app-leaderboard.html)

Rate this Page

icon smile

icon meh

icon frown

Would you like to add a message?

Submit

Thank you for your Feedback!

[](https://ida.interchain.io/ida-course/discord-info.html)

On this page

[What you will be building and why](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html#what-you-will-be-building-and-why)

[Adding a local leaderboard module to the checkers chain](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html#adding-a-local-leaderboard-module-to-the-checkers-chain)

[Unit tests](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html#unit-tests)

[Forwarding player information via IBC](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/6-ibc-app-checkers.html#forwarding-player-information-via-ibc)

#### **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)