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

[Interchain Developer Academy](https://ida.interchain.io/)/[Interchain Developer Academy](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)



Search

[Interchain Developer Academy](https://ida.interchain.io/)[Interchain Developer Academy](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

Search



Filters

Interchain Developer Academy

[](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Week 0 - Getting Started](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Getting Started](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Blockchain 101](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Blockchain History](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Public and Managed Blockchains](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Consensus in Distributed Networks](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Cryptography](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Self-Assessment Quiz](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Go Introduction - First Steps](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Go Basics](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Go Interfaces](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Control Structures in Go](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Arrays and Slices in Go](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Standard Packages in Go](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Concurrency in Go](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Good-To-Know Dev Terms](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Docker Introduction](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Week 1 - Introduction to the Interchain](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Introduction to the Interchain](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Blockchain Technology and the Interchain](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[The Interchain Ecosystem](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Getting ATOM and Staking It](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[A Blockchain App Architecture](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Accounts](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Transactions](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Messages](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Modules](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Protobuf](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Multistore and Keepers](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[BaseApp](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Queries](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Events](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Context](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Testing](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Relaying with IBC](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Interchain Security](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Bridges](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Migrations](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Week 1 Quiz](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Week 2 - First Steps](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[First Steps](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Setup Your Work Environment](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Run a Node, API, and CLI](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Ignite CLI](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Exercise - Make a Checkers Blockchain](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Store Object](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Create Custom Messages](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Create and Save a Game Properly](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Add a Way to Make a Move](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Emit Game Information](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Record the Game Winner](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Week 2 Exercise](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Week 3 - Introduction to IBC and CosmJS](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Introduction to IBC and CosmJS](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[What is IBC?](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[IBC/TAO - Connections (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[IBC/TAO - Channels (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[IBC/TAO - Clients (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[IBC Token Transfer](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Interchain Accounts (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[IBC Middleware (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Create a Custom IBC Middleware (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Integrate IBC Middleware Into a Chain (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[IBC Tooling](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[What is CosmJS?](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Your First CosmJS Actions](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Compose Complex Transactions](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Learn to Integrate Keplr](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Create Custom CosmJS Interfaces](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Week 4 - Ignite CLI and IBC Advanced](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Ignite CLI and IBC Advanced](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Keep an Up-To-Date Game Deadline](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Keep Track Of How Many Moves Have Been Played](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Put Your Games in Order](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Auto-Expiring Games](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Let Players Set a Wager](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Handle wager payments](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Integration tests](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Incentivize Players](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Help Find a Correct Move](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Play With Cross-Chain Tokens](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Understand IBC Denoms](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Go Relayer](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Hermes Relayer](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Week 5 - CosmJS Advanced](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[CosmJS Advanced](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Create Custom Objects](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Create Custom Messages](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Get an External GUI](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Integrate CosmJS and Keplr](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Backend Script for Game Indexing](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Week 6 - IBC Deep Dive](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[IBC Deep Dive](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[IBC Application Developer Introduction](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Make a Module IBC-Enabled](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Adding Packet and Acknowledgment Data](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Extend the Checkers Game With a Leaderboard](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Create a Leaderboard Chain](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Week 7 - From Code to MVP to Production and Migrations](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[From Code to MVP to Production and Migrations](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Run in Production](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Prepare the Software to Run](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Prepare a Validator and Keys](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Prepare Where the Node Starts](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Prepare and Connect to Other Nodes](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Configure, Run, and Set Up a Service](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Prepare and Do Migrations](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Simulate Production in Docker](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Tally Player Info After Production](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Add a Leaderboard as a Module](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Migrate the Leaderboard Module After Production](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Simulate a Migration in Docker](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Final Exam](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[What's Next?](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

[Continue Your Interchain Journey](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html)

Docs Version Switcher

On this page

[Some initial thoughts](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#some-initial-thoughts)

[Code needs](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#code-needs)

[With Ignite CLI](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#with-ignite-cli)

[The message basic validation](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#the-message-basic-validation)

[The move handling](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#the-move-handling)

[Unit tests](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#unit-tests)

[On the message](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#on-the-message)

[On the keeper](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#on-the-keeper)

[Interact via the CLI](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#interact-via-the-cli)

[Bob plays out of turn](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#bob-plays-out-of-turn)

[Alice plays a wrong move](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#alice-plays-a-wrong-move)

[Alice plays correctly](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#alice-plays-correctly)

[#Copy link](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#add-a-way-to-make-a-move) **Add a Way to Make a Move**



Make sure you have all you need before proceeding:

* You understand the concepts of [transactions](https://ida.interchain.io/academy/2-cosmos-concepts/3-transactions.html), [messages](https://ida.interchain.io/academy/2-cosmos-concepts/4-messages.html), and [Protobuf](https://ida.interchain.io/academy/2-cosmos-concepts/6-protobuf.html).
* Go is installed.
* You have the checkers blockchain codebase with MsgCreateGame and its handling. If not, follow the [previous steps](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/5-create-handling.html) or check out the [relevant version (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/tree/create-game-handler).



In this section, you will:

* Extend message handling - play the game.
* Handle moves and update the game state.
* Validate input.
* Extend unit tests.

Your blockchain can now create games, but can you play them? Not yet...so what do you need to make this possible?

[#Copy link](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#some-initial-thoughts) Some initial thoughts

Before diving into the exercise, take some time to think about the following questions:

* What goes into the message?
* How do you sanitize the inputs?
* How do you unequivocally identify games?
* How do you report back errors?
* How do you use your files that implement the checkers rules?
* How do you make sure that nothing is lost?

[#Copy link](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#code-needs) Code needs

When it comes to the code you need, ask yourself:

* What Ignite CLI commands will create your message?
* How do you adjust what Ignite CLI created for you?
* How would you unit-test these new elements?
* How would you use Ignite CLI to locally run a one-node blockchain and interact with it via the CLI to see what you get?

As before, do not bother yet with niceties like gas metering or event emission.

To play a game a player only needs to specify:

* The ID of the game the player wants to join. Call the field gameIndex.
* The initial positions of the pawn. Call the fields fromX and fromY and make them uint.
* The final position of the pawn after a player's move. Call the fields toX and toY to be uint too.

The player does not need to be explicitly added as a field in the message because the player **is** implicitly the signer of the message. Name the object PlayMove.

Unlike when creating the game, you want to return:

* The captured piece, if any. Call the fields capturedX and capturedY. Make then int so that you can pass -1 when no pieces have been captured.
* The (potential) winner in the field winner.

[#Copy link](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#with-ignite-cli) With Ignite CLI

Ignite CLI can create the message and the response objects with a single command:

**Local**

**Docker**



Copy

$ ignite scaffold message playMove gameIndex fromX:uint fromY:uint toX:uint toY:uint \

--module checkers \

--response capturedX:int,capturedY:int,winner

Copy

$ docker run --rm -it \

-v $(pwd):/checkers \

-w /checkers \

checkers\_i \

ignite scaffold message playMove gameIndex fromX:uint fromY:uint toX:uint toY:uint \

--module checkers \

--response capturedX:int,capturedY:int,winner

Ignite CLI once more creates all the necessary Protobuf files and boilerplate for you. See tx.proto:



Copy

message MsgPlayMove {

string creator = 1;

string gameIndex = 2;

uint64 fromX = 3;

uint64 fromY = 4;

uint64 toX = 5;

uint64 toY = 6;

}

message MsgPlayMoveResponse {

int32 capturedX = 1;

int32 capturedY = 2;

string winner = 3;

}

proto /

checkers /

tx.proto

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-msg/proto/checkers/tx.proto" \l "L25-L38" \t "_blank)

All you have to do is fill in the needed parts:

* In x/checkers/types/message\_play\_move.go:



Copy

func (msg \*MsgPlayMove) ValidateBasic() error {

\_, err := sdk.AccAddressFromBech32(msg.Creator)

if err != nil {

return sdkerrors.Wrapf(sdkerrors.ErrInvalidAddress, "invalid creator address (%s)", err)

}

return nil

}

x /

checkers /

types /

message\_play\_move.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-msg/x/checkers/types/message_play_move.go" \l "L44-L50" \t "_blank)

* And in x/checkers/keeper/msg\_server\_play\_move.go:



Copy

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

ctx := sdk.UnwrapSDKContext(goCtx)

// TODO: Handling the message

\_ = ctx

return &types.MsgPlayMoveResponse{}, nil

}

x /

checkers /

keeper /

msg\_server\_play\_move.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-msg/x/checkers/keeper/msg_server_play_move.go" \l "L10-L17" \t "_blank)

Where the TODO is replaced as per the following.

[#Copy link](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#the-message-basic-validation) The message basic validation

With a game index and board positions, there are a number of stateless error situations that can be detected:

* You know there will not be a game index at the value given.
* A piece position is out of the bounds of the board.
* from and to are identical.

Declare your new errors:



Copy

var (

...

+ ErrInvalidGameIndex = sdkerrors.Register(ModuleName, 1103, "game index is invalid")

+ ErrInvalidPositionIndex = sdkerrors.Register(ModuleName, 1104, "position index is invalid")

+ ErrMoveAbsent = sdkerrors.Register(ModuleName, 1105, "there is no move")

)

x /

checkers /

types /

errors.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/types/errors.go" \l "L14-L16" \t "_blank)

Then you can check that:

1. The game index is reasonable:



Copy

gameIndex, err := strconv.ParseInt(msg.GameIndex, 10, 64)

if err != nil {

return sdkerrors.Wrapf(ErrInvalidGameIndex, "not parseable (%s)", err)

}

if uint64(gameIndex) < DefaultIndex {

return sdkerrors.Wrapf(ErrInvalidGameIndex, "number too low (%d)", gameIndex)

}

x /

checkers /

types /

message\_play\_move.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/types/message_play_move.go" \l "L52-L58" \t "_blank)

1. The positions are within bounds, checking an array of situations:



Copy

boardChecks := []struct {

value uint64

err string

}{

{

value: msg.FromX,

err: "fromX out of range (%d)",

},

{

value: msg.ToX,

err: "toX out of range (%d)",

},

{

value: msg.FromY,

err: "fromY out of range (%d)",

},

{

value: msg.ToY,

err: "toY out of range (%d)",

},

}

for \_, situation := range boardChecks {

if situation.value < 0 || rules.BOARD\_DIM <= situation.value {

return sdkerrors.Wrapf(ErrInvalidPositionIndex, situation.err, situation.value)

}

}

x /

checkers /

types /

message\_play\_move.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/types/message_play_move.go" \l "L59-L84" \t "_blank)

Yes, a uint64 like msg.FromY can never be < 0, but since there is no compilation warning you can keep it for future reference if the type changes.

1. There is an actual move:



Copy

if msg.FromX == msg.ToX && msg.FromY == msg.ToY {

return sdkerrors.Wrapf(ErrMoveAbsent, "x (%d) and y (%d)", msg.FromX, msg.FromY)

}

x /

checkers /

types /

message\_play\_move.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/types/message_play_move.go" \l "L85-L87" \t "_blank)



It is conceivable, perhaps even for the benefit of players, to add more stateless checks. For instance, to detect when playing out of wrong cells; after all, only half the cells are valid. Or to detect when moves are not along a diagonal.   
  
These are all worthy checks, although they tend to distract from learning about Cosmos SDK.   
  
If you are nonetheless interested, a good place to look at for a start is the [rules file (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/rules/checkers.go#L66-L69).

[#Copy link](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#the-move-handling) The move handling

The rules represent the ready-made file containing the rules of the game you imported earlier. Declare your new errors in x/checkers/types/errors.go, given your code has to handle new error situations:



Copy

var (

...

+ ErrGameNotFound = sdkerrors.Register(ModuleName, 1106, "game by id not found")

+ ErrCreatorNotPlayer = sdkerrors.Register(ModuleName, 1107, "message creator is not a player")

+ ErrNotPlayerTurn = sdkerrors.Register(ModuleName, 1108, "player tried to play out of turn")

+ ErrWrongMove = sdkerrors.Register(ModuleName, 1109, "wrong move")

)

x /

checkers /

types /

errors.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/types/errors.go" \l "L17-L20" \t "_blank)

Take the following steps to replace the TODO:

1. Fetch the stored game information using the [Keeper.GetStoredGame (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/stored_game.go#L19) function created by Ignite CLI:



Copy

storedGame, found := k.Keeper.GetStoredGame(ctx, msg.GameIndex)

if !found {

return nil, sdkerrors.Wrapf(types.ErrGameNotFound, "%s", msg.GameIndex)

}

x /

checkers /

keeper /

msg\_server\_play\_move.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move.go" \l "L15-L18" \t "_blank)

You return an error because this is a player mistake.

1. Is the player legitimate? Check with:



Copy

isBlack := storedGame.Black == msg.Creator

isRed := storedGame.Red == msg.Creator

var player rules.Player

if !isBlack && !isRed {

return nil, sdkerrors.Wrapf(types.ErrCreatorNotPlayer, "%s", msg.Creator)

} else if isBlack && isRed {

player = rules.StringPieces[storedGame.Turn].Player

} else if isBlack {

player = rules.BLACK\_PLAYER

} else {

player = rules.RED\_PLAYER

}

x /

checkers /

keeper /

msg\_server\_play\_move.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move.go" \l "L20-L31" \t "_blank)

This uses the certainty that the MsgPlayMove.Creator has been verified [by its signature (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/types/message_play_move.go#L29-L35).

1. Instantiate the board in order to implement the rules:



Copy

game, err := storedGame.ParseGame()

if err != nil {

panic(err.Error())

}

x /

checkers /

keeper /

msg\_server\_play\_move.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move.go" \l "L33-L36" \t "_blank)

Fortunately you previously created [this helper (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/types/full_game.go#L22-L32). Here you panic because if the game cannot be parsed the cause may be database corruption.

1. Is it the player's turn? Check using the rules file's own [TurnIs (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/175f467/x/checkers/rules/checkers.go#L145-L147) function:



Copy

if !game.TurnIs(player) {

return nil, sdkerrors.Wrapf(types.ErrNotPlayerTurn, "%s", player)

}

x /

checkers /

keeper /

msg\_server\_play\_move.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move.go" \l "L38-L40" \t "_blank)

1. Properly conduct the move, using the rules' [Move (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/rules/checkers.go#L274-L301) function:



Copy

captured, moveErr := game.Move(

rules.Pos{

X: int(msg.FromX),

Y: int(msg.FromY),

},

rules.Pos{

X: int(msg.ToX),

Y: int(msg.ToY),

},

)

if moveErr != nil {

return nil, sdkerrors.Wrapf(types.ErrWrongMove, moveErr.Error())

}

x /

checkers /

keeper /

msg\_server\_play\_move.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move.go" \l "L42-L54" \t "_blank)

1. Prepare the updated board to be stored and store the information:



Copy

storedGame.Board = game.String()

storedGame.Turn = rules.PieceStrings[game.Turn]

k.Keeper.SetStoredGame(ctx, storedGame)

x /

checkers /

keeper /

msg\_server\_play\_move.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move.go" \l "L56-L58" \t "_blank)

This updates the fields that were modified using the [Keeper.SetStoredGame (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/stored_game.go#L10) function, as when you created and saved the game.

1. Return relevant information regarding the move's result:



Copy

return &types.MsgPlayMoveResponse{

CapturedX: int32(captured.X),

CapturedY: int32(captured.Y),

Winner: rules.PieceStrings[game.Winner()],

}, nil

x /

checkers /

keeper /

msg\_server\_play\_move.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move.go" \l "L60-L64" \t "_blank)

The Captured and Winner information would be lost if you did not get it out of the function one way or another. More accurately, one would have to replay the transaction to discover the values. It is best to make this information easily accessible.

This completes the move process, facilitated by good preparation and the use of Ignite CLI.

[#Copy link](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#unit-tests) Unit tests

Adding unit tests for this play message is very similar to what you did for the previous message.

[#Copy link](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#on-the-message) On the message

Adjust and add to types/message\_play\_move\_test.go. First, change its package for consistency:



Copy

- package types

+ package types\_test

import(

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

)

x /

checkers /

keeper /

msg\_server\_play\_move\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move_test.go" \l "L1-L8" \t "_blank)

Then adjust and add to the test cases:



Collapse



Copy

{

name: "invalid address",

- msg: MsgPlayMove{

+ msg: types.MsgPlayMove{

Creator: "invalid\_address",

+ GameIndex: "5",

+ FromX: 0,

+ FromY: 5,

+ ToX: 1,

+ ToY: 4,

},

err: sdkerrors.ErrInvalidAddress,

},

+ {

+ name: "invalid game index",

+ msg: types.MsgPlayMove{

+ Creator: sample.AccAddress(),

+ GameIndex: "invalid\_index",

+ FromX: 0,

+ FromY: 5,

+ ToX: 1,

+ ToY: 4,

+ },

+ err: types.ErrInvalidGameIndex,

+ },

+ {

+ name: "invalid fromX too high",

+ msg: types.MsgPlayMove{

+ Creator: sample.AccAddress(),

+ GameIndex: "5",

+ FromX: rules.BOARD\_DIM,

+ FromY: 5,

+ ToX: 1,

+ ToY: 4,

+ },

+ err: types.ErrInvalidPositionIndex,

+ },

+ {

+ name: "invalid fromY too high",

+ msg: types.MsgPlayMove{

+ Creator: sample.AccAddress(),

+ GameIndex: "5",

+ FromX: 0,

+ FromY: rules.BOARD\_DIM,

+ ToX: 1,

+ ToY: 4,

+ },

+ err: types.ErrInvalidPositionIndex,

+ },

+ {

+ name: "invalid toX too high",

+ msg: types.MsgPlayMove{

+ Creator: sample.AccAddress(),

+ GameIndex: "5",

+ FromX: 0,

+ FromY: 5,

+ ToX: rules.BOARD\_DIM,

+ ToY: 4,

+ },

+ err: types.ErrInvalidPositionIndex,

+ },

+ {

+ name: "invalid toY too high",

+ msg: types.MsgPlayMove{

+ Creator: sample.AccAddress(),

+ GameIndex: "5",

+ FromX: 0,

+ FromY: 5,

+ ToX: 1,

+ ToY: rules.BOARD\_DIM,

+ },

+ err: types.ErrInvalidPositionIndex,

+ },

+ {

+ name: "invalid no move",

+ msg: types.MsgPlayMove{

+ Creator: sample.AccAddress(),

+ GameIndex: "5",

+ FromX: 0,

+ FromY: 5,

+ ToX: 0,

+ ToY: 5,

+ },

+ err: types.ErrMoveAbsent,

+ },

{

name: "valid address",

- msg: MsgPlayMove{

+ msg: types.MsgPlayMove{

Creator: sample.AccAddress(),

+ GameIndex: "5",

+ FromX: 0,

+ FromY: 5,

+ ToX: 1,

+ ToY: 4,

},

},



x /

checkers /

keeper /

msg\_server\_play\_move\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move_test.go" \l "L19-L113" \t "_blank)

You can try these tests:

**Local**

**Docker**



Copy

$ go test github.com/alice/checkers/x/checkers/types

Copy

$ docker run --rm -it \

-v $(pwd):/checkers \

-w /checkers \

checkers\_i \

go test github.com/alice/checkers/x/checkers/types

[#Copy link](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#on-the-keeper) On the keeper

Create a new keeper/msg\_server\_play\_move\_test.go file and declare it as package keeper\_test. Start with a function that conveniently sets up the keeper for the tests. In this case, already having a game saved can reduce several lines of code in each test:



Copy

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

k, ctx := keepertest.CheckersKeeper(t)

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

server := keeper.NewMsgServerImpl(\*k)

context := sdk.WrapSDKContext(ctx)

server.CreateGame(context, &types.MsgCreateGame{

Creator: alice,

Black: bob,

Red: carol,

})

return server, \*k, context

}

x /

checkers /

keeper /

msg\_server\_play\_move\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move_test.go" \l "L15-L26" \t "_blank)

Note that it reuses alice, bob and carol found in the file [msg\_server\_create\_game\_test.go (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_create_game_test.go#L16-L18) of the same package.

Now test the result of a move. Blacks play first, which according to setupMsgServerWithOneGameForPlayMove corresponds to bob:



Copy

func TestPlayMove(t \*testing.T) {

msgServer, \_, context := setupMsgServerWithOneGameForPlayMove(t)

playMoveResponse, err := msgServer.PlayMove(context, &types.MsgPlayMove{

Creator: bob,

GameIndex: "1",

FromX: 1,

FromY: 2,

ToX: 2,

ToY: 3,

})

require.Nil(t, err)

require.EqualValues(t, types.MsgPlayMoveResponse{

CapturedX: -1,

CapturedY: -1,

Winner: "\*",

}, \*playMoveResponse)

}

x /

checkers /

keeper /

msg\_server\_play\_move\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move_test.go" \l "L28-L44" \t "_blank)

Also test whether the game was [saved correctly (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move_test.go#L83-L108). Check what happens when [the game cannot be found (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move_test.go#L46-L58), the sender [is not a player (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move_test.go#L110-L122), a player tries to [play out of turn (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move_test.go#L145-L157), or [makes a wrong move (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move_test.go#L159-L171). Check after [two (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move_test.go#L173-L232) or [three turns with a capture (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move_test.go#L234-L309).

As a special case, add a test to check what happens when a board is not parseable, which is expected to end up in a panic, not with a returned error:



Copy

func TestPlayMoveCannotParseGame(t \*testing.T) {

msgServer, k, context := setupMsgServerWithOneGameForPlayMove(t)

ctx := sdk.UnwrapSDKContext(context)

storedGame, \_ := k.GetStoredGame(ctx, "1")

storedGame.Board = "not a board"

k.SetStoredGame(ctx, storedGame)

defer func() {

r := recover()

require.NotNil(t, r, "The code did not panic")

require.Equal(t, r, "game cannot be parsed: invalid board string: not a board")

}()

msgServer.PlayMove(context, &types.MsgPlayMove{

Creator: bob,

GameIndex: "1",

FromX: 1,

FromY: 2,

ToX: 2,

ToY: 3,

})

}

x /

checkers /

keeper /

msg\_server\_play\_move\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/play-move-handler/x/checkers/keeper/msg_server_play_move_test.go" \l "L124-L143" \t "_blank)

Note the use of [defer (opens new window)↗](https://stackoverflow.com/a/31596110), which can be used as a Go way of implementing try catch of panics. The defer statement is set up right before the msgServer.PlayMove statement that is expected to fail, so that it does not *catch* panics that may happen earlier.

Try these tests:

**Local**

**Docker**



Copy

$ go test github.com/alice/checkers/x/checkers/keeper

Copy

$ docker run --rm -it \

-v $(pwd):/checkers \

-w /checkers \

checkers\_i \

go test github.com/alice/checkers/x/checkers/keeper

[#Copy link](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#interact-via-the-cli) Interact via the CLI

Start your chain again:

**Local**

**Docker**



Copy

$ ignite chain serve

Copy

$ docker run --rm -it \

--name checkers \

-v $(pwd):/checkers \

-w /checkers \

checkers\_i \

ignite chain serve

If you restarted from the [previous section](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/5-create-handling.html), there is already one game in storage and it is waiting for Alice's move. If that is not the case, recreate a game via the CLI.

[#Copy link](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#bob-plays-out-of-turn) Bob plays out of turn

Can Bob make a move? Look at the play-move message and which parameters it expects:

**Local**

**Docker**



Copy

$ checkersd tx checkers play-move --help

Copy

$ docker exec -it checkers \

checkersd tx checkers play-move --help

This returns:



Copy

Broadcast message playMove

Usage:

checkersd tx checkers play-move [game-index] [from-x] [from-y] [to-x] [to-y] [flags]

...

So Bob tries:

**Local**

**Docker**



Copy

$ checkersd tx checkers play-move 1 0 5 1 4 --from $bob

^ ^ ^ ^ ^

| | | | To Y

| | | To X

| | From Y

| From X

Game id

Copy

$ docker exec -it checkers \

checkersd tx checkers play-move 1 0 5 1 4 --from $bob

^ ^ ^ ^ ^

| | | | To Y

| | | To X

| | From Y

| From X

Game id

After you accept sending the transaction, it should complain with the result including:



Copy

...

raw\_log: 'failed to execute message; message index: 0: {red}: player tried to play

out of turn'

...

txhash: D10BB8A706870F65F19E4DF48FB870E4B7D55AF4232AE0F6897C23466FF7871B



If you did not get this raw\_log, your transaction may have been sent asynchronously. You can always query a transaction by using the txhash with the following command:

**Local**

**Docker**



Copy

$ checkersd query tx D10BB8A706870F65F19E4DF48FB870E4B7D55AF4232AE0F6897C23466FF7871B

Copy

$ docker exec -it checkers \

checkersd query tx D10BB8A706870F65F19E4DF48FB870E4B7D55AF4232AE0F6897C23466FF7871B

And you are back on track:



Copy

...

raw\_log: 'failed to execute message; message index: 0: {red}: player tried to play

out of turn'

This error by Bob was caught when he tried to play out of turn. The check was a *stateful* check as the message itself was valid. This failure cost him gas.

[#Copy link](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#alice-plays-a-wrong-move) Alice plays a wrong move

Can Alice, who plays *black*, make a move? Can she make a wrong move? There are two kinds of wrong moves that Alice can make: she can make one whose wrongness will be caught statelessly, and another that will be caught because of the current state of the board.

1. As an example of a *statelessly wrong* move, she could try to take a piece on the side and move it just outside the board:

**Local**

**Docker**



Copy

$ checkersd tx checkers play-move 1 7 2 8 3 --from $alice

Copy

$ docker exec -it checkers \

checkersd tx checkers play-move 1 7 2 8 3 --from $alice

The computer says "no" immediately:



Copy

Error: toX out of range (8): position index is invalid

...

The transaction never went into the mem pool. This mistake did not cost Alice any gas.

1. As an example of a *statefully wrong* move, Alice can try to move from 0-1 to 1-0, which is occupied by one of her pieces.

**Local**

**Docker**



Copy

$ checkersd tx checkers play-move 1 1 0 0 1 --from $alice

Copy

$ docker exec -it checkers \

checkersd tx checkers play-move 1 1 0 0 1 --from $alice

The computer says "no" again, but this time after the transaction has been validated:



Copy

...

raw\_log: 'failed to execute message; message index: 0: Already piece at destination

position: {0 1}: wrong move'

This mistake cost Alice some gas.

So far all seems to be working.

[#Copy link](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#alice-plays-correctly) Alice plays correctly

Time for Alice to make a correct move:

**Local**

**Docker**



Copy

$ checkersd tx checkers play-move 1 1 2 2 3 --from $alice

Copy

$ docker exec -it checkers \

checkersd tx checkers play-move 1 1 2 2 3 --from $alice

This returns:



Copy

...

raw\_log: '[{"events":[{"type":"message","attributes":[{"key":"action","value":"play\_move"}]}]}]'

Confirm the move went through with your one-line formatter from the [previous section](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/5-create-handling.html):

**Linux**

**Docker**

**Mac**



Copy

$ checkersd query checkers show-stored-game 1 --output json | jq ".storedGame.board" | sed 's/"//g' | sed 's/|/\n/g'

Copy

$ docker exec -it checkers \

bash -c "checkersd query checkers show-stored-game 1 --output json | jq \".storedGame.board\" | sed 's/\"//g' | sed 's/|/\n/g'"

Copy

$ checkersd query checkers show-stored-game 1 --output json | jq ".storedGame.board" | sed 's/"//g' | sed 's/|/\'$'\n/g'

This shows:



Copy

\*b\*b\*b\*b

b\*b\*b\*b\*

\*\*\*b\*b\*b

\*\*b\*\*\*\*\* <--- Here

\*\*\*\*\*\*\*\*

r\*r\*r\*r\*

\*r\*r\*r\*r

r\*r\*r\*r\*

Alice's piece moved down and right.

When you are done with this exercise you can stop Ignite's chain serve.

synopsis

To summarize, this section has explored:

* How to add stateless checks on your message.
* How to use messages and handlers, in this case to add the capability of actually playing moves on checkers games created in your application.
* The information that needs to be specified for a game move message to function, which are the game ID, the initial positions of the pawn to be moved, and the final positions of the pawn at the end of the move.
* The information necessary to return, which includes the game ID, the location of any captured piece, and the registration of a winner should the game be won as a result of the move.
* How to modify the response object created by Ignite CLI to add additional fields.
* How to implement and check the steps required by move handling, including the declaration of the ready-made rules in the errors.go file so your code can handle new error situations.
* How to add unit tests to check the functionality of your code.
* How to interact via the CLI to confirm that correct player turn order is enforced by the application.

previous

[](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/5-create-handling.html)

**[Create and Save a Game Properly](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/5-create-handling.html)**

up next

**[Emit Game Information](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/7-events.html)**

[[](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/7-events.html)](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/7-events.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

[Some initial thoughts](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#some-initial-thoughts)

[Code needs](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#code-needs)

[With Ignite CLI](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#with-ignite-cli)

[The message basic validation](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#the-message-basic-validation)

[The move handling](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#the-move-handling)

[Unit tests](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#unit-tests)

[On the message](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#on-the-message)

[On the keeper](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#on-the-keeper)

[Interact via the CLI](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#interact-via-the-cli)

[Bob plays out of turn](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#bob-plays-out-of-turn)

[Alice plays a wrong move](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#alice-plays-a-wrong-move)

[Alice plays correctly](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/6-play-game.html#alice-plays-correctly)

#### **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)