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

[Interchain Developer Academy](https://ida.interchain.io/)/[Interchain Developer Academy](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)



Search

[Interchain Developer Academy](https://ida.interchain.io/)[Interchain Developer Academy](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

Search



Filters

Interchain Developer Academy

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Week 0 - Getting Started](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Getting Started](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Blockchain 101](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Blockchain History](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Public and Managed Blockchains](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Consensus in Distributed Networks](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Cryptography](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Self-Assessment Quiz](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Go Introduction - First Steps](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Go Basics](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Go Interfaces](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Control Structures in Go](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Arrays and Slices in Go](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Standard Packages in Go](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Concurrency in Go](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Good-To-Know Dev Terms](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Docker Introduction](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Week 1 - Introduction to the Interchain](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Introduction to the Interchain](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Blockchain Technology and the Interchain](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[The Interchain Ecosystem](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Getting ATOM and Staking It](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[A Blockchain App Architecture](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Accounts](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Transactions](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Messages](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Modules](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Protobuf](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Multistore and Keepers](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[BaseApp](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Queries](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Events](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Context](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Testing](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Relaying with IBC](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Interchain Security](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Bridges](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Migrations](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Week 1 Quiz](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Week 2 - First Steps](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[First Steps](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Setup Your Work Environment](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Run a Node, API, and CLI](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Ignite CLI](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Exercise - Make a Checkers Blockchain](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Store Object](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Create Custom Messages](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Create and Save a Game Properly](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Add a Way to Make a Move](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Emit Game Information](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Record the Game Winner](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Week 2 Exercise](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Week 3 - Introduction to IBC and CosmJS](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Introduction to IBC and CosmJS](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[What is IBC?](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[IBC/TAO - Connections (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[IBC/TAO - Channels (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[IBC/TAO - Clients (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[IBC Token Transfer](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Interchain Accounts (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[IBC Middleware (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Create a Custom IBC Middleware (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Integrate IBC Middleware Into a Chain (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[IBC Tooling](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[What is CosmJS?](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Your First CosmJS Actions](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Compose Complex Transactions](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Learn to Integrate Keplr](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Create Custom CosmJS Interfaces](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Week 4 - Ignite CLI and IBC Advanced](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Ignite CLI and IBC Advanced](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Keep an Up-To-Date Game Deadline](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Keep Track Of How Many Moves Have Been Played](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Put Your Games in Order](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Auto-Expiring Games](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Let Players Set a Wager](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Handle wager payments](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Integration tests](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Incentivize Players](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Help Find a Correct Move](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Play With Cross-Chain Tokens](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Understand IBC Denoms](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Go Relayer](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Hermes Relayer](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Week 5 - CosmJS Advanced](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[CosmJS Advanced](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Create Custom Objects](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Create Custom Messages](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Get an External GUI](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Integrate CosmJS and Keplr](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Backend Script for Game Indexing](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Week 6 - IBC Deep Dive](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[IBC Deep Dive](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[IBC Application Developer Introduction](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Make a Module IBC-Enabled](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Adding Packet and Acknowledgment Data](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Extend the Checkers Game With a Leaderboard](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Create a Leaderboard Chain](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Week 7 - From Code to MVP to Production and Migrations](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[From Code to MVP to Production and Migrations](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Run in Production](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Prepare the Software to Run](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Prepare a Validator and Keys](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Prepare Where the Node Starts](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Prepare and Connect to Other Nodes](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Configure, Run, and Set Up a Service](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Prepare and Do Migrations](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Simulate Production in Docker](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Tally Player Info After Production](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Add a Leaderboard as a Module](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Migrate the Leaderboard Module After Production](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Simulate a Migration in Docker](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Final Exam](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[What's Next?](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

[Continue Your Interchain Journey](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html)

Docs Version Switcher

On this page

[Some initial thoughts](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#some-initial-thoughts)

[Code needs](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#code-needs)

[Why would you get rid of stale games?](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#why-would-you-get-rid-of-stale-games)

[How can you get rid of expired games?](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#how-can-you-get-rid-of-expired-games)

[New information](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#new-information)

[FIFO management](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#fifo-management)

[FIFO Integration](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#fifo-integration)

[Unit tests](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#unit-tests)

[Interact via the CLI](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#interact-via-the-cli)

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#put-your-games-in-order) **Put Your Games in Order**



Make sure you have everything you need before proceeding:

* You understand the concepts of [ABCI](https://ida.interchain.io/academy/2-cosmos-concepts/1-architecture.html), [Protobuf](https://ida.interchain.io/academy/2-cosmos-concepts/6-protobuf.html), and of a [doubly-linked list (opens new window)↗](https://en.wikipedia.org/wiki/Doubly_linked_list).
* Go is installed.
* You have the checkers blockchain codebase with Winner, GameDeadline, and MoveCount. If not, follow the [previous steps](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/2-move-count.html) or check out [the relevant version (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/tree/move-count).



In this section, you will deal with:

* A FIFO data structure
* FIFO unit tests

You will learn:

* Modularity and data organization styles

In previous steps, you added the [game winner](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/8-game-winner.html), the [game deadline](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/1-game-deadline.html), and the [move count](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/2-move-count.html). These are fields that can help you determine whether a given game is stale or not.

Specifically:

* If the game has a winner, then it is not considered stale.
* If the game's deadline is in the future, then it is not stale.
* If the game's count is 0 or 1 then it should be deleted when stale.
* If the game's count is 2 or more, then it should be forfeited when stale.

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#some-initial-thoughts) Some initial thoughts

The key difficulty is to get the list of games that satisfy the *staleness* criteria without grinding your blockchain to a halt with an *O(n)* call, where *n* is the total number of games in storage.

A solution is to introduce a new data structure that will let you find one stale game in *O(1)*, and access all *k* stale games in *O(k)*.

There are some initial thoughts and code needs to keep in mind during the next sections to be able to implement forfeits in the end.

Before you begin touching your code, ask:

* What conditions have to be satisfied for a game to be considered stale and the blockchain to act? There are some pointers in the first paragraph.
* How would you get rid of stale games as part of the protocol, that is *without user inputs*?
* How do you optimize performance and data structures so that a few stale games do not cause your blockchain to grind to a halt?
* How can you be sure that your blockchain is safe from attacks?
* How do you make your changes compatible with future plans for wagers?
* Are there errors to report back?
* What event should you emit?

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#code-needs) Code needs

Now, think about what possible code changes and additions you should consider:

* What Ignite CLI commands, if any, will assist you?
* 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?

For now, do not bother yet with future ideas like wager handling.

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#why-would-you-get-rid-of-stale-games) Why would you get rid of stale games?

There is only one way for a game to advance through its lifecycle until resolution, win or draw: to *play* it.

Game inactivity could become a factor. What if a player never shows up again? Should a game remain in limbo forever?

Eventually you want to let players wager on the outcome of games, so you do not want games remaining in limbo if they have *value* assigned. For this reason, you need a way for games to be forcibly resolved if one player stops responding.



The simplest mechanism to expire a game is to use a **deadline**. If the deadline is reached, then the game is forcibly terminated and expires. The deadline is pushed back every time a move is played.

To enforce the termination, it is a good idea to use the **EndBlock** part of the ABCI protocol. The call EndBlock is triggered when all transactions of the block are delivered, and allows you to tidy up before the block is sealed. In your case, all games that have reached their deadline will be terminated.

How do you find all the games that have reached their deadline? You could use a pseudo-code like:



Copy

findAll(game => game.deadline < now)

This approach is **expensive** in terms of computation. The EndBlock code should not have to pull up all games out of the storage just to find a few that are relevant. Doing a findAll costs [O(n) (opens new window)↗](https://en.wikipedia.org/wiki/Big_O_notation), where n is the total number of games.

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#how-can-you-get-rid-of-expired-games) How can you get rid of expired games?

You need another data structure. The simplest option is a First-In-First-Out (FIFO) that is constantly updated, so that:

* When games are played, they are taken out of where they are and sent to the tail.
* It follows that games that have not been played for the longest time eventually rise to the head.

Therefore, when terminating expired games in EndBlock, you deal with the expired games that are at the head of the FIFO. You do not stop until the head includes an ongoing game. The cost is:

* O(1) on each game creation and gameplay.
* O(k) where k is the number of expired games on each block.
* k <= n where n is the number of games that exist.

k is still an unbounded number of operations. However, if you use the same expiration duration on each game, for k games to expire together in a block they would all have to have had a move in the same previous block (give or take the block before or after). In the worst case, the largest EndBlock computation will be proportional to the largest regular block in the past. This is a reasonable risk to take.



This only works if the expiration duration is the same for all games, instead of being a parameter left to a potentially malicious game creator.

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#new-information) New information

How do you implement a FIFO from which you extract elements at random positions? Choose a doubly-linked list:

1. You must remember the game ID at the head to pick expired games, and at the tail to send back fresh games. The existing SystemInfo object is useful, as it is already expandable. Add to its Protobuf declaration:



Copy

message SystemInfo {

...

+ string fifoHeadIndex = 2; // Will contain the index of the game at the head.

+ string fifoTailIndex = 3; // Will contain the index of the game at the tail.

}

proto /

checkers /

system\_info.proto

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/fifo-fields/proto/checkers/system_info.proto" \l "L8-L9" \t "_blank)

1. To make extraction possible, each game must know which other game takes place before it in the FIFO, and which after. Store this double-link information in StoredGame. Add them to the game's Protobuf declaration:



Copy

message StoredGame {

...

+ string beforeIndex = 9; // Pertains to the FIFO. Toward head.

+ string afterIndex = 10; // Pertains to the FIFO. Toward tail.

}

proto /

checkers /

stored\_game.proto

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/fifo-fields/proto/checkers/stored_game.proto" \l "L15-L16" \t "_blank)

1. There must be an "ID" that indicates *no game*. Use "-1", which you save as a constant:



Copy

const (

NoFifoIndex = "-1"

)

x /

checkers /

types /

keys.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/fifo-fields/x/checkers/types/keys.go" \l "L53-L55" \t "_blank)

1. Instruct Ignite CLI and Protobuf to regenerate the Protobuf files:

**Local**

**Docker**



Copy

$ ignite generate proto-go

Copy

$ docker run --rm -it \

-v $(pwd):/checkers \

-w /checkers \

checkers\_i \

ignite generate proto-go

1. Adjust the default genesis values, so that it has a proper head and tail:



Copy

func DefaultGenesis() \*GenesisState {

return &GenesisState{

SystemInfo: SystemInfo{

NextId: uint64(DefaultIndex),

+ FifoHeadIndex: NoFifoIndex,

+ FifoTailIndex: NoFifoIndex,

},

...

}

}

x /

checkers /

types /

genesis.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/fifo-fields/x/checkers/types/genesis.go" \l "L15-L16" \t "_blank)

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#fifo-management) FIFO management

Now that the new fields are created, you need to update them to keep your FIFO up-to-date. It's better to create a separate file that encapsulates this knowledge. Create x/checkers/keeper/stored\_game\_in\_fifo.go with the following:

1. A function to remove from the FIFO:



Copy

func (k Keeper) RemoveFromFifo(ctx sdk.Context, game \*types.StoredGame, info \*types.SystemInfo) {

// Does it have a predecessor?

if game.BeforeIndex != types.NoFifoIndex {

beforeElement, found := k.GetStoredGame(ctx, game.BeforeIndex)

if !found {

panic("Element before in Fifo was not found")

}

beforeElement.AfterIndex = game.AfterIndex

k.SetStoredGame(ctx, beforeElement)

if game.AfterIndex == types.NoFifoIndex {

info.FifoTailIndex = beforeElement.Index

}

// Is it at the FIFO head?

} else if info.FifoHeadIndex == game.Index {

info.FifoHeadIndex = game.AfterIndex

}

// Does it have a successor?

if game.AfterIndex != types.NoFifoIndex {

afterElement, found := k.GetStoredGame(ctx, game.AfterIndex)

if !found {

panic("Element after in Fifo was not found")

}

afterElement.BeforeIndex = game.BeforeIndex

k.SetStoredGame(ctx, afterElement)

if game.BeforeIndex == types.NoFifoIndex {

info.FifoHeadIndex = afterElement.Index

}

// Is it at the FIFO tail?

} else if info.FifoTailIndex == game.Index {

info.FifoTailIndex = game.BeforeIndex

}

game.BeforeIndex = types.NoFifoIndex

game.AfterIndex = types.NoFifoIndex

}

x /

checkers /

keeper /

stored\_game\_in\_fifo.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/game-fifo/x/checkers/keeper/stored_game_in_fifo.go" \l "L8-L41" \t "_blank)

The game passed as an argument is **not** saved in storage here, even if it was updated. Only its fields in memory are adjusted. The *before* and *after* games are saved in storage. Do a SetStoredGame after calling this function to avoid having a mix of saves and memory states. The same applies to SetSystemInfo.

1. A function to send to the tail:



Copy

func (k Keeper) SendToFifoTail(ctx sdk.Context, game \*types.StoredGame, info \*types.SystemInfo) {

if info.FifoHeadIndex == types.NoFifoIndex && info.FifoTailIndex == types.NoFifoIndex {

game.BeforeIndex = types.NoFifoIndex

game.AfterIndex = types.NoFifoIndex

info.FifoHeadIndex = game.Index

info.FifoTailIndex = game.Index

} else if info.FifoHeadIndex == types.NoFifoIndex || info.FifoTailIndex == types.NoFifoIndex {

panic("Fifo should have both head and tail or none")

} else if info.FifoTailIndex == game.Index {

// Nothing to do, already at tail

} else {

// Snip game out

k.RemoveFromFifo(ctx, game, info)

// Now add to tail

currentTail, found := k.GetStoredGame(ctx, info.FifoTailIndex)

if !found {

panic("Current Fifo tail was not found")

}

currentTail.AfterIndex = game.Index

k.SetStoredGame(ctx, currentTail)

game.BeforeIndex = currentTail.Index

info.FifoTailIndex = game.Index

}

}

x /

checkers /

keeper /

stored\_game\_in\_fifo.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/game-fifo/x/checkers/keeper/stored_game_in_fifo.go" \l "L43-L68" \t "_blank)

Again, it is advisable to do SetStoredGame and SetSystemInfo after calling this function.

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#fifo-integration) FIFO Integration

With these functions ready, it is time to use them in the message handlers.

1. In the handler when creating a new game, set default values for BeforeIndex and AfterIndex:



Copy

...

storedGame := types.StoredGame{

...

+ BeforeIndex: types.NoFifoIndex,

+ AfterIndex: types.NoFifoIndex,

}

x /

checkers /

keeper /

msg\_server\_create\_game.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/fifo-fields/x/checkers/keeper/msg_server_create_game.go" \l "L31-L32" \t "_blank)

Send the new game to the tail because it is freshly created:



Copy

...

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

k.Keeper.SetStoredGame(ctx, storedGame)

...

x /

checkers /

keeper /

msg\_server\_create\_game.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/game-fifo/x/checkers/keeper/msg_server_create_game.go" \l "L40" \t "_blank)

1. In the handler, when playing a move send the game back to the tail because it was freshly updated (unless it was won, in which case it has to be removed):



Copy

...

lastBoard := game.String()

+

+ systemInfo, found := k.Keeper.GetSystemInfo(ctx)

+ if !found {

+ panic("SystemInfo not found")

+ }

+

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

storedGame.Board = lastBoard

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

} else {

storedGame.Board = ""

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

}

...

k.Keeper.SetStoredGame(ctx, storedGame)

+ k.Keeper.SetSystemInfo(ctx, systemInfo)

...

x /

checkers /

keeper /

msg\_server\_play\_move.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/game-fifo/x/checkers/keeper/msg_server_play_move.go" \l "L65-L82" \t "_blank)

Note that you also need to call SetSystemInfo.

You have implemented a FIFO that is updated but never really used. It will be used in the [next section](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/4-game-forfeit.html).

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#unit-tests) Unit tests

At this point, your previous unit tests are failing, so they must be fixed. Add FifoHeadIndex and FifoTailIndex in your value requirements on SystemInfo as you [create games (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/game-fifo/x/checkers/keeper/msg_server_create_game_test.go#L46-L47) and [play moves (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/game-fifo/x/checkers/keeper/msg_server_play_move_test.go#L98-L99). Also add BeforeIndex and AfterIndex in your value requirements on StoredGame as you [create games (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/game-fifo/x/checkers/keeper/msg_server_create_game_test.go#L60-L61) and [play moves (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/game-fifo/x/checkers/keeper/msg_server_play_move_test.go#L112-L113).

Edit types/genesis\_test.go, so that it incorporates the newly added fields FifoHeadIndex and FifoTailIndex with the correct initialization values:



Copy

func TestDefaultGenesisState\_ExpectedInitialNextId(t \*testing.T) {

require.EqualValues(t,

&types.GenesisState{

StoredGameList: []types.StoredGame{},

- SystemInfo: types.SystemInfo{uint64(1)},

+ SystemInfo: types.SystemInfo{

+ NextId: uint64(1),

+ FifoHeadIndex: "-1",

+ FifoTailIndex: "-1",

+ },

},

types.DefaultGenesis())

}

x /

checkers /

types /

genesis\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/fifo-fields/x/checkers/types/genesis_test.go" \l "L71-L75" \t "_blank)

Next, you should add more specific FIFO tests. For instance, testing what happens to SystemInfo and StoredGame as you create up to three new games, instead of [only checking at the end (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/game-fifo/x/checkers/keeper/msg_server_create_game_test.go#L165-L232):



Copy

func TestCreate3GamesHasSavedFifo(t \*testing.T) {

msgSrvr, keeper, context := setupMsgServerCreateGame(t)

ctx := sdk.UnwrapSDKContext(context)

msgSrvr.CreateGame(context, &types.MsgCreateGame{

Creator: alice,

Black: bob,

Red: carol,

})

// Second game

msgSrvr.CreateGame(context, &types.MsgCreateGame{

Creator: bob,

Black: carol,

Red: alice,

})

systemInfo2, found := keeper.GetSystemInfo(ctx)

require.True(t, found)

require.EqualValues(t, types.SystemInfo{

NextId: 3,

FifoHeadIndex: "1",

FifoTailIndex: "2",

}, systemInfo2)

game1, found := keeper.GetStoredGame(ctx, "1")

require.True(t, found)

require.EqualValues(t, types.StoredGame{

Index: "1",

Board: "\*b\*b\*b\*b|b\*b\*b\*b\*|\*b\*b\*b\*b|\*\*\*\*\*\*\*\*|\*\*\*\*\*\*\*\*|r\*r\*r\*r\*|\*r\*r\*r\*r|r\*r\*r\*r\*",

Turn: "b",

Black: bob,

Red: carol,

Winner: "\*",

Deadline: types.FormatDeadline(ctx.BlockTime().Add(types.MaxTurnDuration)),

MoveCount: uint64(0),

BeforeIndex: "-1",

AfterIndex: "2",

}, game1)

game2, found := keeper.GetStoredGame(ctx, "2")

require.True(t, found)

require.EqualValues(t, types.StoredGame{

Index: "2",

Board: "\*b\*b\*b\*b|b\*b\*b\*b\*|\*b\*b\*b\*b|\*\*\*\*\*\*\*\*|\*\*\*\*\*\*\*\*|r\*r\*r\*r\*|\*r\*r\*r\*r|r\*r\*r\*r\*",

Turn: "b",

Black: carol,

Red: alice,

Winner: "\*",

Deadline: types.FormatDeadline(ctx.BlockTime().Add(types.MaxTurnDuration)),

MoveCount: uint64(0),

BeforeIndex: "1",

AfterIndex: "-1",

}, game2)

// Third game

msgSrvr.CreateGame(context, &types.MsgCreateGame{

Creator: carol,

Black: alice,

Red: bob,

})

systemInfo3, found := keeper.GetSystemInfo(ctx)

require.True(t, found)

require.EqualValues(t, types.SystemInfo{

NextId: 4,

FifoHeadIndex: "1",

FifoTailIndex: "3",

}, systemInfo3)

game1, found = keeper.GetStoredGame(ctx, "1")

require.True(t, found)

require.EqualValues(t, types.StoredGame{

Index: "1",

Board: "\*b\*b\*b\*b|b\*b\*b\*b\*|\*b\*b\*b\*b|\*\*\*\*\*\*\*\*|\*\*\*\*\*\*\*\*|r\*r\*r\*r\*|\*r\*r\*r\*r|r\*r\*r\*r\*",

Turn: "b",

Black: bob,

Red: carol,

Winner: "\*",

Deadline: types.FormatDeadline(ctx.BlockTime().Add(types.MaxTurnDuration)),

MoveCount: uint64(0),

BeforeIndex: "-1",

AfterIndex: "2",

}, game1)

game2, found = keeper.GetStoredGame(ctx, "2")

require.True(t, found)

require.EqualValues(t, types.StoredGame{

Index: "2",

Board: "\*b\*b\*b\*b|b\*b\*b\*b\*|\*b\*b\*b\*b|\*\*\*\*\*\*\*\*|\*\*\*\*\*\*\*\*|r\*r\*r\*r\*|\*r\*r\*r\*r|r\*r\*r\*r\*",

Turn: "b",

Black: carol,

Red: alice,

Winner: "\*",

Deadline: types.FormatDeadline(ctx.BlockTime().Add(types.MaxTurnDuration)),

MoveCount: uint64(0),

BeforeIndex: "1",

AfterIndex: "3",

}, game2)

game3, found := keeper.GetStoredGame(ctx, "3")

require.True(t, found)

require.EqualValues(t, types.StoredGame{

Index: "3",

Board: "\*b\*b\*b\*b|b\*b\*b\*b\*|\*b\*b\*b\*b|\*\*\*\*\*\*\*\*|\*\*\*\*\*\*\*\*|r\*r\*r\*r\*|\*r\*r\*r\*r|r\*r\*r\*r\*",

Turn: "b",

Black: alice,

Red: bob,

Winner: "\*",

Deadline: types.FormatDeadline(ctx.BlockTime().Add(types.MaxTurnDuration)),

MoveCount: uint64(0),

BeforeIndex: "2",

AfterIndex: "-1",

}, game3)

}

Expand



x /

checkers /

keeper /

msg\_server\_create\_game\_fifo\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/game-fifo/x/checkers/keeper/msg_server_create_game_fifo_test.go" \l "L11-L117" \t "_blank)

What happens when you [have two games and play once on the *older* one (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/game-fifo/x/checkers/keeper/msg_server_play_move_fifo_test.go#L11-L63)? Or have two games and play them twice in turn:



Copy

func TestPlayMove2Games2MovesHasSavedFifo(t \*testing.T) {

msgServer, keeper, context := setupMsgServerWithOneGameForPlayMove(t)

ctx := sdk.UnwrapSDKContext(context)

msgServer.CreateGame(context, &types.MsgCreateGame{

Creator: bob,

Black: carol,

Red: alice,

})

msgServer.PlayMove(context, &types.MsgPlayMove{

Creator: bob,

GameIndex: "1",

FromX: 1,

FromY: 2,

ToX: 2,

ToY: 3,

})

msgServer.PlayMove(context, &types.MsgPlayMove{

Creator: carol,

GameIndex: "2",

FromX: 1,

FromY: 2,

ToX: 2,

ToY: 3,

})

systemInfo1, found := keeper.GetSystemInfo(ctx)

require.True(t, found)

require.EqualValues(t, types.SystemInfo{

NextId: 3,

FifoHeadIndex: "1",

FifoTailIndex: "2",

}, systemInfo1)

game1, found := keeper.GetStoredGame(ctx, "1")

require.True(t, found)

require.EqualValues(t, types.StoredGame{

Index: "1",

Board: "\*b\*b\*b\*b|b\*b\*b\*b\*|\*\*\*b\*b\*b|\*\*b\*\*\*\*\*|\*\*\*\*\*\*\*\*|r\*r\*r\*r\*|\*r\*r\*r\*r|r\*r\*r\*r\*",

Turn: "r",

Black: bob,

Red: carol,

Winner: "\*",

Deadline: types.FormatDeadline(ctx.BlockTime().Add(types.MaxTurnDuration)),

MoveCount: uint64(1),

BeforeIndex: "-1",

AfterIndex: "2",

}, game1)

game2, found := keeper.GetStoredGame(ctx, "2")

require.True(t, found)

require.EqualValues(t, types.StoredGame{

Index: "2",

Board: "\*b\*b\*b\*b|b\*b\*b\*b\*|\*\*\*b\*b\*b|\*\*b\*\*\*\*\*|\*\*\*\*\*\*\*\*|r\*r\*r\*r\*|\*r\*r\*r\*r|r\*r\*r\*r\*",

Turn: "r",

Black: carol,

Red: alice,

Winner: "\*",

Deadline: types.FormatDeadline(ctx.BlockTime().Add(types.MaxTurnDuration)),

MoveCount: uint64(1),

BeforeIndex: "1",

AfterIndex: "-1",

}, game2)

}

Expand



x /

checkers /

keeper /

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

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/game-fifo/x/checkers/keeper/msg_server_play_move_fifo_test.go" \l "L65-L125" \t "_blank)

Run the tests again with the same command as before:

**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/2-ignite-cli-adv/3-game-fifo.html#interact-via-the-cli) Interact via the CLI

Time to explore the commands. You need to start afresh because you made numerous additions to the blockchain state:

**Local**

**Docker**



Copy

$ ignite chain serve --reset-once

Copy

$ docker run --rm -it \

--name checkers \

-v $(pwd):/checkers \

-w /checkers \

checkers\_i \

ignite chain serve --reset-once



Do not forget to export alice and bob again, as explained in an [earlier section under "Interact via the CLI"](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/4-create-message.html).

1

Is the genesis FIFO information correctly saved?

**Local**

**Docker**



Copy

$ checkersd query checkers show-system-info

Copy

$ docker exec -it checkers \

checkersd query checkers show-system-info

This should print:



Copy

SystemInfo:

fifoHeadIndex: "-1" # There is nothing

fifoTailIndex: "-1" # There is nothing

nextId: "1"

2

If you create a game, is the game as expected?

**Local**

**Docker**



Copy

$ checkersd tx checkers create-game $alice $bob --from $bob

$ checkersd query checkers show-system-info

Copy

$ docker exec -it checkers \

checkersd tx checkers create-game $alice $bob --from $bob

$ docker exec -it checkers \

checkersd query checkers show-system-info

This should print:



Copy

SystemInfo:

fifoHeadIndex: "1" # The first game you created

fifoTailIndex: "1" # The first game you created

nextId: "2"

3

What about the information saved in the game?

**Local**

**Docker**



Copy

$ checkersd query checkers show-stored-game 1

Copy

$ docker exec -it checkers \

checkersd query checkers show-stored-game 1

Because it is the only game, this should print:



Copy

storedGame:

afterIndex: "-1" # Nothing because it is alone

beforeIndex: "-1" # Nothing because it is alone

...

4

And if you create another game?

**Local**

**Docker**



Copy

$ checkersd tx checkers create-game $alice $bob --from $bob

$ checkersd query checkers show-system-info

Copy

$ docker exec -it checkers \

checkersd tx checkers create-game $alice $bob --from $bob

$ docker exec -it checkers \

checkersd query checkers show-system-info

This should print:



Copy

SystemInfo:

fifoHeadIndex: "1" # The first game you created

fifoTailIndex: "2" # The second game you created

nextId: "3"

5

Did the games also store the correct values?

For the first game:

**Local**

**Docker**



Copy

$ checkersd query checkers show-stored-game 1

Copy

$ docker exec -it checkers \

checkersd query checkers show-stored-game 1

This should print:



Copy

afterIndex: "2" # The second game you created

beforeIndex: "-1" # No game

...

For the second game, run:

**Local**

**Docker**



Copy

$ checkersd query checkers show-stored-game 2

Copy

$ docker exec -it checkers \

checkersd query checkers show-stored-game 2

This should print:



Copy

afterIndex: "-1" # No game

beforeIndex: "1" # The first game you created

...

Your FIFO in effect has the game IDs [1, 2].

Add a third game, and confirm that your FIFO is [1, 2, 3].

6

What happens if Alice plays a move in game 2, the game *in the middle*?

**Local**

**Docker**



Copy

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

$ checkersd query checkers show-system-info

Copy

$ docker exec -it checkers \

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

$ docker exec -it checkers \

checkersd query checkers show-system-info

This should print:



Copy

SystemInfo:

fifoHeadIndex: "1" # The first game you created

fifoTailIndex: "2" # The second game you created and on which Bob just played

nextId: "4"

7

Is game 3 in the middle now?

**Local**

**Docker**



Copy

$ checkersd query checkers show-stored-game 3

Copy

$ docker exec -it checkers \

checkersd query checkers show-stored-game 3

This should print:



Copy

storedGame:

afterIndex: "2"

beforeIndex: "1"

...

Your FIFO now has the game IDs [1, 3, 2]. You see that game 2, which was played on, has been sent to the tail of the FIFO.

Which is what you were expecting.

synopsis

To summarize, this section has explored:

* The use of a First-In-First-Out (FIFO) data structure to sort games from the least recently played at the top of the list to the most recently played at the bottom, in order to help identify inactive games which may become candidates for forced termination, which reduces undesirable and wasteful data stored on the blockchain.
* How forced termination of games is beneficial should you implement a wager system, as it prevents any assigned value from becoming locked into inactive games by causing the inactive player to forfeit the game and lose their wager.
* How any code solution which searches the entire data store for inactive games is computationally expensive, needlessly accessing many active games to identify any inactive minority (which may not even exist).
* How a FIFO data structure definitionally orders games such that inactive games rise to the top of the list, meaning code solutions can simply run until encountering the first active game and then stop, conserving gas fees.
* What new information and functions need to be added to your code; how to integrate them into the message handlers; how to update your unit tests to prevent them from failing due to these changes; and what tests to run to test the code.
* How to interact with the CLI to check the effectiveness of your new commands.

previous

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/2-move-count.html)

**[Keep Track Of How Many Moves Have Been Played](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/2-move-count.html)**

up next

**[Auto-Expiring Games](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/4-game-forfeit.html)**

[[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/4-game-forfeit.html)](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/4-game-forfeit.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

[Some initial thoughts](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#some-initial-thoughts)

[Code needs](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#code-needs)

[Why would you get rid of stale games?](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#why-would-you-get-rid-of-stale-games)

[How can you get rid of expired games?](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#how-can-you-get-rid-of-expired-games)

[New information](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#new-information)

[FIFO management](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#fifo-management)

[FIFO Integration](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#fifo-integration)

[Unit tests](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#unit-tests)

[Interact via the CLI](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/3-game-fifo.html#interact-via-the-cli)

#### **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)