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

[Interchain Developer Academy](https://ida.interchain.io/)/[Interchain Developer Academy](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)



Search

[Interchain Developer Academy](https://ida.interchain.io/)[Interchain Developer Academy](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

Search



Filters

Interchain Developer Academy

[](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Week 0 - Getting Started](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Getting Started](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Blockchain 101](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Blockchain History](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Public and Managed Blockchains](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Consensus in Distributed Networks](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Cryptography](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Self-Assessment Quiz](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Go Introduction - First Steps](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Go Basics](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Go Interfaces](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Control Structures in Go](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Arrays and Slices in Go](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Standard Packages in Go](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Concurrency in Go](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Good-To-Know Dev Terms](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Docker Introduction](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Week 1 - Introduction to the Interchain](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Introduction to the Interchain](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Blockchain Technology and the Interchain](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[The Interchain Ecosystem](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Getting ATOM and Staking It](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[A Blockchain App Architecture](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Accounts](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Transactions](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Messages](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Modules](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Protobuf](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Multistore and Keepers](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[BaseApp](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Queries](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Events](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Context](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Testing](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Relaying with IBC](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Interchain Security](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Bridges](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Migrations](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Week 1 Quiz](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Week 2 - First Steps](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[First Steps](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Setup Your Work Environment](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Run a Node, API, and CLI](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Ignite CLI](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Exercise - Make a Checkers Blockchain](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Store Object](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Create Custom Messages](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Create and Save a Game Properly](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Add a Way to Make a Move](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Emit Game Information](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Record the Game Winner](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Week 2 Exercise](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Week 3 - Introduction to IBC and CosmJS](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Introduction to IBC and CosmJS](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[What is IBC?](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[IBC/TAO - Connections (OPTIONAL)](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[IBC/TAO - Channels (OPTIONAL)](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[IBC/TAO - Clients (OPTIONAL)](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[IBC Token Transfer](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Interchain Accounts (OPTIONAL)](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[IBC Middleware (OPTIONAL)](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Create a Custom IBC Middleware (OPTIONAL)](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Integrate IBC Middleware Into a Chain (OPTIONAL)](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[IBC Tooling](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[What is CosmJS?](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Your First CosmJS Actions](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Compose Complex Transactions](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Learn to Integrate Keplr](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Create Custom CosmJS Interfaces](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Week 4 - Ignite CLI and IBC Advanced](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Ignite CLI and IBC Advanced](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Keep an Up-To-Date Game Deadline](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Keep Track Of How Many Moves Have Been Played](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Put Your Games in Order](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Auto-Expiring Games](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Let Players Set a Wager](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Handle wager payments](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Integration tests](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Incentivize Players](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Help Find a Correct Move](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Play With Cross-Chain Tokens](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Understand IBC Denoms](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Go Relayer](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Hermes Relayer](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Week 5 - CosmJS Advanced](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[CosmJS Advanced](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Create Custom Objects](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Create Custom Messages](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Get an External GUI](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Integrate CosmJS and Keplr](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Backend Script for Game Indexing](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Week 6 - IBC Deep Dive](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[IBC Deep Dive](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[IBC Application Developer Introduction](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Make a Module IBC-Enabled](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Adding Packet and Acknowledgment Data](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Extend the Checkers Game With a Leaderboard](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Create a Leaderboard Chain](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Week 7 - From Code to MVP to Production and Migrations](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[From Code to MVP to Production and Migrations](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Run in Production](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Prepare the Software to Run](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Prepare a Validator and Keys](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Prepare Where the Node Starts](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Prepare and Connect to Other Nodes](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Configure, Run, and Set Up a Service](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Prepare and Do Migrations](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Simulate Production in Docker](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Tally Player Info After Production](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Add a Leaderboard as a Module](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Migrate the Leaderboard Module After Production](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Simulate a Migration in Docker](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Final Exam](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[What's Next?](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

[Continue Your Interchain Journey](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html)

Docs Version Switcher

On this page

[Process overview](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#process-overview)

[Plan](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#plan)

[Sidecar process](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#sidecar-process)

[UpgradeHandler](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#upgradehandler)

[StoreLoader](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#storeloader)

[Proposal](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#proposal)

[Advantages](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#advantages)

[Effect of upgrades](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#effect-of-upgrades)

[Application-specific](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#application-specific)

[Cosmovisor](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#cosmovisor)

[Code example](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#code-example)

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#migrations) **Migrations**



To better understand this section, first read the following sections:

* [Messages](https://ida.interchain.io/academy/2-cosmos-concepts/4-messages.html)
* [Protobuf](https://ida.interchain.io/academy/2-cosmos-concepts/6-protobuf.html)



Have you ever wondered how an upgrade is done in the Cosmos SDK? In this section you will find out how Cosmos SDK migrations are conducted.   
  
Blockchains can be upgraded through a predictable process that reliably avoids forks. Discover the Interchain's comprehensive process that includes governance, data migrations, node upgrades, and more, to ensure upgrades proceed smoothly and without service disruption.   
  
At the end of the section, the code example demonstrates how you would use migration to upgrade your checkers blockchain with new features even after it has been in operation for some time.

A Cosmos SDK application running on a Cosmos blockchain can be upgraded in an orderly, on-chain fashion.



Upgrading blockchains and blockchain applications is notoriously difficult and risky. The Cosmos SDK solves the common risks and challenges.

Generally, when a blockchain is upgraded it is vital that all nodes upgrade simultaneously and at the same block height. This is difficult to achieve in a disorderly setting. If the nodes do not coordinate, then the blockchain will "fork" into two blockchains with a common history: one chain that observes the new rules, and one chain that observes the old rules. It is generally not possible for these two chains to reach a common consensus or merge in the future.



Upgrading a live chain without software support for upgrades is risky, because all the validators need to pause their state machines at the same block height and apply the upgrade before resuming. If this is not done correctly, there can be state inconsistencies, which are hard to recover from.



Smart contracts on EVM chains such as Ethereum are immutable software. They are difficult or impossible to change by definition. Various strategies based on modularity can simulate the effects of upgrading smart contracts, but all known methods have inherent limitations, in particular the difficulties, impossibility, or prohibitive costs of re-organizing data at rest. This places a significant limitation on the types of upgrades that are feasible.

**A Cosmos SDK blockchain built for a specific application can be upgraded without forks.** If a new version of the blockchain application uses a different data layout than exists on the chain, the existing data can be reorganized before the new version of the application comes online. The data migration is defined by the developer and runs in each node quickly and cost-effectively before the node returns to service.

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#process-overview) Process overview

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#plan) Plan

A "plan" is an upgrade process to take place at a specific block height in the future. It includes a SideCar process that executes when the upgrade commences, which names the plan and specifies a block height at which to execute.



Acceptance or rejection of the plan is managed through the normal governance process. A "cancel proposal" can be submitted and adopted, preventing the plan from executing. Cancellation is contingent on knowing that a given plan is a poor idea before the upgrade happens.

The Info in a plan kicks off the SideCar process:



Copy

type Plan struct {

Name string

Height int64

Info string

}

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#sidecar-process) Sidecar process

A SideCar is a binary which nodes can run to attend to processes outside of Cosmos binaries. This can include steps such as downloading and compiling software from a certain commit in a repo.

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#upgradehandler) UpgradeHandler

An UpgradeHandler may be executed after the SideCar process is finished and the binary has been upgraded. It attends to on-chain activities that may be necessary before normal processing resumes. An upgrade handler may trigger a StoreLoader.

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#storeloader) StoreLoader

A StoreLoader prepares the on-chain state for use by the new binary. This can include reorganizing existing data. The node does not resume normal operation until the store loader has returned and the handler has completed its work.

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#proposal) Proposal

Governance uses proposals that are voted on and adopted or rejected. An upgrade proposal takes the form of accepting or rejecting a plan that is prepared and submitted through governance. Proposals can be withdrawn before execution with cancellation proposals.

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#advantages) Advantages

Coordinated upgrades attend to the challenging process of upgrading blockchain applications and blockchain platforms.

The main advantages of this form of coordinated upgrades are:

* **Avoidance of forks:** all validators move together at a pre-determined block height.
* **Smooth upgrade of binaries:** the new software is adopted in an automated fashion.
* **Reorganizing data stores:** data at rest can be reorganized as needed by processes that are not limited by factors such as a block gas limit.

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#effect-of-upgrades) Effect of upgrades

Blockchains are paused at the block height of an adopted plan. This initiates the upgrade process. The upgrade process itself may include switching to a new binary that is relatively small to download and install, or it may include an extensive data reorganization process. The validator stops processing blocks until it completes the process in either case.

The validator resumes processing blocks when the handler is satisfied with the completeness degree of the upgrade. From a user perspective, this appears as a pause which resumes with the new version.

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#application-specific) Application-specific

The SideCar, handler, and store loader are application-specific. At each block, the Cosmos SDK checks for a plan that should be executed before processing block transactions. If none exists, then processing continues as usual. If a plan is scheduled to run, then the Cosmos SDK pauses normal processing and loads the SideCar. When the SideCar is finished, it loads the handler and optionally the store loader.

Application developers build implementations of those components that are tailored to their application and use case.



For a more detailed explanation of the upgrade process, see the [Cosmos SDK documentation (opens new window)↗](https://docs.cosmos.network/v0.46/modules/upgrade).

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#cosmovisor) Cosmovisor

Cosmovisor is a tool that node operators can use to automate the on-chain processes described above.

* Cosmovisor runs as a small wrapper around the Cosmos SDK application binaries.
* Cosmovisor watches for any approved upgrade proposals.
* Cosmovisor can download and run the new binary if wanted.
* When the chain reaches the upgrade block, Cosmovisor also handles the storage upgrade.



See the [Cosmos SDK documentation on Cosmovisor (opens new window)↗](https://docs.cosmos.network/main/tooling/cosmovisor) to learn more about this process manager for Cosmos SDK applications.

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#code-example) Code example

****

**Show me some code for my checkers blockchain**

The code samples you have seen previously were meant to build your checkers blockchain from the ground up. Now imagine it has been running in production for some time, with games created, played on, won, and lost. Given its success and following player feedback, you decide to introduce leaderboards. In particular:

* Any player who has **ever** played should have a tally of games won, lost, drawn, and forfeited.
* There should be a leaderboard that lists the players with the most wins, but in limited numbers (for instance, only the top 100 scores).
* To increase engagement, the player with the most recent score takes precedence over an *older* contender with an equal score.

It is not good enough to introduce a leaderboard for players currently winning and losing: you want to start with **all** those that played in the past. Fortunately, you have kept all past games and their outcomes in the chain state. What you need to do is go through the record, update the players with their tallies, and add a leaderboard. Call your existing app version **v1**. To disambiguate, call your new one with the leaderboard **v2** and the upgrade's name "v1tov2". **New information**

You need new data structures for v2. With Ignite CLI you have:

1. A way to store each player's information:

Copy

message PlayerInfo {

string index = 1; // The stringified address

uint64 wonCount = 2;

uint64 lostCount = 3;

uint64 forfeitedCount = 4;

}

1. A way to identify a winning player's information in the leaderboard:

Copy

message WinningPlayer {

string playerAddress = 1;

uint64 wonCount = 2;

string dateAdded = 3;

}

wonCount decides the position on the leaderboard, and dateAdded resolves ties for equal wonCounts, with the most recent ranking higher.

1. Now you need a leaderboard as an array of winner information:

Copy

import "checkers/winning\_player.proto";

message Leaderboard {

repeated WinningPlayer winners = 1;

}

1. Because this player information and this leaderboard are stored somewhere in storage, you must introduce them in the new genesis:

Copy

message GenesisState {

...

repeated PlayerInfo playerInfoList = 3;

Leaderboard leaderboard = 4;

}

Conceptually, it is a hypothetical *new* v2 genesis because your actual genesis file did not contain any leaderboard.

1. Finally, you must set a hard-coded leaderboard length:

Copy

const (

LeaderboardWinnerLength = 100

)

**Leaderboard on-the-go updating**

Before thinking about the upgrade, take care of the code as if your v2 was a new project. You need to add code to your v2 to update the leaderboard after a game has been determined. This means a lot of array sorting and information adjustment on the previous code.

Look at the links at the bottom of this page for the related hands-on guided exercise.

**Genesis migration preparation**

With on-the-go updating of the leaderboard taken care of in v2, you must place past players on the leaderboard. You choose the **in-place migration**, whereby your v2 software has access to the v1 storage the first time it launches and *migrates* it to a v2 storage as fast as it can. **Past player handling**

Now prepare functions to progressively build the player's information, given a list of games. To improve performance, you can choose to use [Go routines (opens new window)](https://gobyexample.com/goroutines) and [channels (opens new window)](https://gobyexample.com/channels) so that the in-memory computation can proceed on the current data chunk while the next data chunk is being fetched from storage, in a manner reminiscent of map/reduce. Without going into too much detail, the following actions are taken:

* Games are read from storage 1,000 at a time.
* A Go routine computes the intermediate pieces of player information and then passes them on.
* These intermediate pieces are added to the player information totals from storage.

Again, look for links at the bottom of this page for the detailed implementation.

**Past leaderboard**

Eventually, the player information computation is complete and it is possible to create the leaderboard for these past players. This may involve the sorting of a very large array. Perhaps it could be done in tranches:

Copy

const (

// Adjust this length to obtain the best performance over a large map.

PlayerInfoChunkSize = types.LeaderboardWinnerLength \* 2

)

Then, you use a Go routine and channel again such that:

* Player information is read from storage chunks at a time.
* The intermediate leaderboard is combined with this chunk and sorted again.

With this done, you can encapsulate in a new function the order in which the state migration takes place:

Copy

func PerformMigration(ctx sdk.Context, k keeper.Keeper, storedGameChunk uint64, playerInfoChunk uint64) error {

err := MapStoredGamesReduceToPlayerInfo(ctx, k, storedGameChunk)

if err != nil {

return err

}

err = MapPlayerInfosReduceToLeaderboard(ctx, k, playerInfoChunk)

if err != nil {

return err

}

return nil

}

For details about the implementation, look for the links at the bottom of this page.

**Proper genesis migration**

With the data migration prepared, it is time to:

1. Inform your module about its consensus versions.
2. Inform the app about its upgrade versions.

Make explicit your module's new ConsensusVersion. It should increment strictly. If you previously had 2, you can increment it to 3:

Copy

func (AppModule) ConsensusVersion() uint64 { return 3 }

With that, the upgrade module knows it has to look for migration information to go from 2 to 3. If you had put 4, the upgrade module would know it has to look for migration information to go from 2 to 3 and then from 3 to 4.

Have your module inform the app about what it has to do when it encounters the old 2 version:

Copy

func (am AppModule) RegisterServices(cfg module.Configurator) {

...

if err := cfg.RegisterMigration(types.ModuleName, "2", func(ctx sdk.Context) error {

return v1tov2.PerformMigration(ctx, am.keeper, v1tov2.StoredGameChunkSize, v1tov2.PlayerInfoChunkSize)

}); err != nil {

panic(fmt.Errorf("failed to register migration of %s to v2: %w", types.ModuleName, err))

}

}

If you had put 4, you would have to add another if ... "3" - not else if ... "3". With the module informed about what it has to do to migrate its state from one consensus version to the next, you need to inform the app how to handle the whole app version, from v1 to v2. Such an app upgrade could cover a state migration for more than one module. The app already calls your module's RegisterServices, so you do not need to add anything here. If it is not already the case, make sure your app has a Configurator:

Copy

type App struct {

...

configurator module.Configurator

}

Ensure that the configurator is populated:

Copy

app.configurator = module.NewConfigurator(app.appCodec, app.MsgServiceRouter(), app.GRPCQueryRouter())

app.mm.RegisterServices(app.configurator)

Then create a function that encapsulates what needs to be done when encountering the "v1tov2" upgrade name:

Copy

func (app \*App) setupUpgradeHandlers() {

// v1 to v2 upgrade handler

app.UpgradeKeeper.SetUpgradeHandler(

"v1tov2",

func(ctx sdk.Context, plan upgradetypes.Plan, vm module.VersionMap) (module.VersionMap, error) {

return app.mm.RunMigrations(ctx, app.configurator, vm)

},

)

...

}

app.mm.RunMigrations will call all the module's state migrations. Finally, make sure your app calls this new function:

Copy

...

app.SetEndBlocker(app.EndBlocker)

app.setupUpgradeHandlers()

if loadLatest {

...

}

StoredGameList and SystemInfo are unchanged from v1 to v2. Also note that all past players are saved with now, since the time was not saved in the game when winning. If you decide to use the Deadline, make sure that there are no times in the future.

The migration mechanism helps identify how you can upgrade your blockchain to introduce new features.



If you want more details on how to update the leaderboard, look at [Running Your Own Cosmos Chain](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/).   
  
For even more detail, see [Tally Player Info After Production](https://ida.interchain.io/hands-on-exercise/4-run-in-prod/2-migration-info.html).

synopsis

To summarize, this section has explored:

* How Cosmos SDK migrations provide developers with an orderly, on-chain process for upgrading their applications, reliably avoiding forks through the use of a "plan" in which upgrades are proposed to occur at a specific future block height.
* How consensus over accepting or rejecting a plan is reached through the normal governance process, ensuring unity across all nodes, as is any "cancel proposal" intended to prevent a previously accepted plan from executing.
* How the temporary halting of normal activity allows for simultaneous and potentially profound modifications of the blockchain across all nodes, including the reorganization of existing data stores to maintain compatibility with the upgraded application.

previous

[](https://ida.interchain.io/academy/2-cosmos-concepts/15-bridges.html)

**[Bridges](https://ida.interchain.io/academy/2-cosmos-concepts/15-bridges.html)**

up next

**[Week 1 Quiz](https://ida.interchain.io/ida-course/quiz-week1.html)**

[[](https://ida.interchain.io/ida-course/quiz-week1.html)](https://ida.interchain.io/ida-course/quiz-week1.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

[Process overview](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#process-overview)

[Plan](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#plan)

[Sidecar process](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#sidecar-process)

[UpgradeHandler](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#upgradehandler)

[StoreLoader](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#storeloader)

[Proposal](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#proposal)

[Advantages](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#advantages)

[Effect of upgrades](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#effect-of-upgrades)

[Application-specific](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#application-specific)

[Cosmovisor](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#cosmovisor)

[Code example](https://ida.interchain.io/academy/2-cosmos-concepts/16-migrations.html#code-example)

#### **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)