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

[Interchain Developer Academy](https://ida.interchain.io/)/[Interchain Developer Academy](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)



Search

[Interchain Developer Academy](https://ida.interchain.io/)[Interchain Developer Academy](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

Search



Filters

Interchain Developer Academy

[](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Week 0 - Getting Started](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Getting Started](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Blockchain 101](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Blockchain History](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Public and Managed Blockchains](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Consensus in Distributed Networks](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Cryptography](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Self-Assessment Quiz](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Go Introduction - First Steps](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Go Basics](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Go Interfaces](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Control Structures in Go](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Arrays and Slices in Go](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Standard Packages in Go](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Concurrency in Go](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Good-To-Know Dev Terms](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Docker Introduction](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Week 1 - Introduction to the Interchain](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Introduction to the Interchain](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Blockchain Technology and the Interchain](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[The Interchain Ecosystem](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Getting ATOM and Staking It](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[A Blockchain App Architecture](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Accounts](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Transactions](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Messages](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Modules](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Protobuf](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Multistore and Keepers](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[BaseApp](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Queries](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Events](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Context](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Testing](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Relaying with IBC](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Interchain Security](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Bridges](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Migrations](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Week 1 Quiz](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Week 2 - First Steps](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[First Steps](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Setup Your Work Environment](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Run a Node, API, and CLI](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Ignite CLI](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Exercise - Make a Checkers Blockchain](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Store Object](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Create Custom Messages](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Create and Save a Game Properly](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Add a Way to Make a Move](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Emit Game Information](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Record the Game Winner](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Week 2 Exercise](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Week 3 - Introduction to IBC and CosmJS](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Introduction to IBC and CosmJS](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[What is IBC?](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[IBC/TAO - Connections (OPTIONAL)](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[IBC/TAO - Channels (OPTIONAL)](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[IBC/TAO - Clients (OPTIONAL)](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[IBC Token Transfer](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Interchain Accounts (OPTIONAL)](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[IBC Middleware (OPTIONAL)](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Create a Custom IBC Middleware (OPTIONAL)](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Integrate IBC Middleware Into a Chain (OPTIONAL)](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[IBC Tooling](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[What is CosmJS?](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Your First CosmJS Actions](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Compose Complex Transactions](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Learn to Integrate Keplr](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Create Custom CosmJS Interfaces](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Week 4 - Ignite CLI and IBC Advanced](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Ignite CLI and IBC Advanced](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Keep an Up-To-Date Game Deadline](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Keep Track Of How Many Moves Have Been Played](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Put Your Games in Order](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Auto-Expiring Games](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Let Players Set a Wager](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Handle wager payments](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Integration tests](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Incentivize Players](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Help Find a Correct Move](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Play With Cross-Chain Tokens](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Understand IBC Denoms](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Go Relayer](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Hermes Relayer](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Week 5 - CosmJS Advanced](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[CosmJS Advanced](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Create Custom Objects](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Create Custom Messages](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Get an External GUI](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Integrate CosmJS and Keplr](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Backend Script for Game Indexing](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Week 6 - IBC Deep Dive](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[IBC Deep Dive](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[IBC Application Developer Introduction](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Make a Module IBC-Enabled](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Adding Packet and Acknowledgment Data](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Extend the Checkers Game With a Leaderboard](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Create a Leaderboard Chain](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Week 7 - From Code to MVP to Production and Migrations](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[From Code to MVP to Production and Migrations](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Run in Production](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Prepare the Software to Run](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Prepare a Validator and Keys](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Prepare Where the Node Starts](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Prepare and Connect to Other Nodes](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Configure, Run, and Set Up a Service](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Prepare and Do Migrations](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Simulate Production in Docker](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Tally Player Info After Production](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Add a Leaderboard as a Module](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Migrate the Leaderboard Module After Production](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Simulate a Migration in Docker](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Final Exam](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[What's Next?](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

[Continue Your Interchain Journey](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html)

Docs Version Switcher

On this page

[Testing pyramid](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html#testing-pyramid)

[Unit tests](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html#unit-tests)

[Integration tests](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html#integration-tests)

[Simulation tests](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html#simulation-tests)

[End-to-end tests](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html#end-to-end-tests)

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html#testing) **Testing**



It is recommended to first read the following sections to better understand testing:

* [Transactions](https://ida.interchain.io/academy/2-cosmos-concepts/3-transactions.html)
* [Messages](https://ida.interchain.io/academy/2-cosmos-concepts/4-messages.html)
* [Modules](https://ida.interchain.io/academy/2-cosmos-concepts/5-modules.html)
* [BaseApp](https://ida.interchain.io/academy/2-cosmos-concepts/8-base-app.html)



Testing is the general umbrella that describes all that revolves around making sure that any code created does what it is expected to do.

Testing is an integral part of software development in general. Done well, it helps teams catch bugs earlier, and thus, repairing problems becomes cheaper. In the context of blockchains, which can lock large amounts of value and where [migrations](https://ida.interchain.io/academy/2-cosmos-concepts/13-migrations.html) are not simply one redeploy away from a fix, bugs can be catastrophic.

The Cosmos SDK implements its own testing vision for its modules, and it would be good if your project followed the same patterns. Following the same patterns will help everyone in the ecosystem *speak the same language*. Speaking the same language is also beneficial when you open the code for a bug bounty. Indeed, readable tests increase the trust that casual observers have in your code overall and, by extension, your project, and allow interested bounty hunters to be *onboarded* faster.

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html#testing-pyramid) Testing pyramid

After some [reflection (opens new window)↗](https://docs.cosmos.network/main/architecture/adr-059-test-scopes.html), the Cosmos SDK divides tests into four broad categories of somewhat increasing scope:

* [Unit tests (opens new window)↗](https://docs.cosmos.network/main/building-modules/testing#unit-tests)
* [Integration tests (opens new window)↗](https://docs.cosmos.network/main/building-modules/testing#integration-tests)
* [Simulation tests (opens new window)↗](https://docs.cosmos.network/main/building-modules/testing#simulations)
* [End-to-end tests (E2E) (opens new window)↗](https://docs.cosmos.network/main/building-modules/testing#end-to-end-tests)

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html#unit-tests) Unit tests

Unit tests are your classic "smallest tests possible", and focus on a single module at a time. If a tested call needs something from another module, like a keeper, then:

* This dependency should be [mocked (opens new window)↗](https://devopedia.org/mock-testing), including mocked responses when applicable.
* The mock should confirm that the dependency was called as expected.
* If applicable, the test should confirm that a mocked response was handled as expected by the module under test.

To be considered well-tested, your unit tests should cover **all** your module's functions.

As an example, imagine that your module moves tokens on behalf of your users. Your module, therefore, has a dependency on the bank keeper. As part of a unit test setup, you create a mocked bank keeper and use it. After the test action, at verification time, your unit test confirms that your module called the expected functions of the bank keeper within the expected parameters. Your unit test does **not** test whether bank balances have changed because, remember, your module does not have a real bank keeper.

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html#integration-tests) Integration tests

Integration tests are one step wider in scope. They are still focused on your module but, instead of mocking the dependencies, now your test provisions a **minimum-viable application** that includes fully-fledged dependencies, including – crucially – those your own module needs.



A minimum-viable application contains your module and all its dependencies, as well as their dependencies, but nothing more.

In a well-designed testing environment, providing such fully-fledged dependencies should not be a concern of your module's tests. All the more so if your dependencies have dependencies of their own. You want to minimize such deep correlations between modules, even in regard to their tests.

This is why, to minimize correlations, from version 0.47 of the Cosmos SDK onward each module exposes functions to provide a minimum viable module. This way, your module only knows how to instantiate itself given fully-fledged dependencies, the *inputs*. An added benefit is that your module exposes explicitly the inputs it needs to instantiate.

Your integration tests start by creating an app that instantiates the list of explicitly defined inputs required. For instance, when creating a minimum-viable app to integration-test the bank module, no slashing module is provisioned, as slashing is a side concern instead of a dependency. Of course, each module requires a different minimum-viable app. To facilitate the creation of such an app and therefore, of integration tests, the Cosmos SDK team has also developed an in-house dependency injection.

To fit in the context of Go testing, modules provide testing suites that encapsulate the test instantiations.



In Cosmos SDK version 0.46 and earlier, what are called "integration tests" are really full tests, where a full application is being instantiated. The reason behind this is that in these versions the coding effort to create a minimum-viable app was not commensurate with the benefit it provided compared to a full app.

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html#simulation-tests) Simulation tests

The purpose of simulation tests is to introduce some random effects into the parameters passed.

Simulation tests are similar in scope to integration tests, where they reuse your module's minimum-viable application. This scope also only starts to make sense from Cosmos SDK 0.47, where the application dependencies are disentangled.

[#Copy link](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html#end-to-end-tests) End-to-end tests

End-to-end (E2E) tests are at the top of the testing pyramid. Unlike integration and simulation tests, they work by testing the *full* application, not a minimum-viable one.

Your E2E tests should test flows that mirror what users would experience, and therefore, should not limit themselves to minute interactions. Conceptually, they are for your whole application, and not per module.



As stated, if you work with Cosmos SDK versions 0.46 or earlier, any tests that are labeled "integration tests" are actually E2E tests under the new designation from version 0.47 onward.

In the context of Go, just like for integration tests, you provide testing suites that handle the instantiations. Ideally, your testing suite should be usable outside of your application so that other applications can test interactions with yours.



**Further reading**

* [Test scopes decision report (opens new window)↗](https://docs.cosmos.network/main/architecture/adr-059-test-scopes.html)

****

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

You can introduce tests as soon as you have created code of your own. You ought not to add tests that specifically focus on code generated by Ignite CLI.

***Simple unit testing***

For instance, when unit-testing the QueryCanPlayMoveRequest handling you could:

* Define game situations, a.k.a. test cases:

Copy

canPlayTestRange = []canPlayGameCase{

{

desc: "First move by black",

game: 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",

Winner: "\*",

},

request: &types.QueryCanPlayMoveRequest{

GameIndex: "1",

Player: "b",

FromX: 1,

FromY: 2,

ToX: 2,

ToY: 3,

},

response: canPlayOkResponse,

err: "nil",

},

...

}

* Use the helper created by Ignite to instantiate a test checkers keeper:

Copy

keeper, ctx := keepertest.CheckersKeeper(t)

* Set up the keeper with the game described in the case:

Copy

keeper.SetStoredGame(ctx, testCase.game)

* Then run the test case:

Copy

response, err := keeper.CanPlayMove(goCtx, testCase.request)

* Then confirm the expectations:

Copy

if testCase.response == nil {

require.Nil(t, response)

} else {

require.EqualValues(t, testCase.response, response)

}

if testCase.err == "nil" {

require.Nil(t, err)

} else {

require.EqualError(t, err, testCase.err)

}

***Unit testing with mocks***

If you need to mock another module's keeper, with the help of [gomock (opens new window)](https://github.com/golang/mock) you would:

* Perform the one-time creation of the mock types of your expected keeper interfaces, which includes the bank keeper.
* Then, for each test, create a mock controller which acts as the puppeteer of your mocked bank keeper:

Copy

ctrl := gomock.NewController(t)

bankMock := testutil.NewMockBankEscrowKeeper(ctrl)

* Initialize your test checkers keeper with the necessary mocks:

Copy

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

* Instruct the mock controller to check all call expectations when the test ends:

Copy

defer ctrl.Finish()

This code is placed before the lines below, but it gets [executed at the end (opens new window)](https://go.dev/tour/flowcontrol/12).

* Add your call expectations on the mock. For instance, here you expect:
  + Alice to put amount in escrow.
  + Bob to do the same, but **after** Alice.
  + No other call to the bank to take place.

Copy

alicePay := escrow.EXPECT().SendCoinsFromAccountToModule(

sdk.UnwrapSDKContext(context),

aliceAddr,

types.ModuleName,

sdk.Coins{

sdk.Coin{

Denom: denom,

Amount: sdk.NewInt(int64(amount)),

},

}

).Times(1)

escrow.EXPECT().SendCoinsFromAccountToModule(

sdk.UnwrapSDKContext(context),

aliceAddr,

types.ModuleName,

sdk.Coins{

sdk.Coin{

Denom: denom,

Amount: sdk.NewInt(int64(amount)),

},

}

).Times(1).After(alicePay)

* Alternatively, remove all expectations by accepting all calls; for instance, if you are testing something else, like game moves:

Copy

bankMock.EXPECT().SendCoinsFromAccountToModule(

sdk.UnwrapSDKContext(context),

gomock.Any(),

gomock.Any(),

gomock.Any()

).AnyTimes()

***Integration testing***

Integration testing is a bit more involved. See the links below.



If you want to go beyond out-of-context code samples like the above and see more in detail how to define these features, go to any section of [Run Your Own Cosmos Chain](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/).   
  
More precisely, you can jump to:

* [Store Object - Make a Checkers Blockchain](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/3-stored-game.html) for the very first, somewhat pointless, unit test.
* [Create Custom Messages](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/4-create-message.html) for the first **unit test** of a message delivery.
* [Handle Wager Payments](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/6-payment-winning.html) where:
  + **Mocking** is introduced so as to mock the bank keeper.
* [Integration Tests](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html) where:
  + The first **integration tests** are introduced, to test the integration with the bank keeper.
* [Tally Player Info](https://ida.interchain.io/hands-on-exercise/4-run-in-prod/2-migration-info.html) where an in-place store migration is unit and integration tested.

synopsis

To summarize, this section has explored:

* Why testing is important.
* How the Cosmos SDK conceptually divides its tests.
* What is the scope and what happens in each test category.

previous

[](https://ida.interchain.io/academy/2-cosmos-concepts/11-context.html)

**[Context](https://ida.interchain.io/academy/2-cosmos-concepts/11-context.html)**

up next

**[Relaying with IBC](https://ida.interchain.io/academy/2-cosmos-concepts/13-relayer-intro.html)**

[[](https://ida.interchain.io/academy/2-cosmos-concepts/13-relayer-intro.html)](https://ida.interchain.io/academy/2-cosmos-concepts/13-relayer-intro.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

[Testing pyramid](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html#testing-pyramid)

[Unit tests](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html#unit-tests)

[Integration tests](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html#integration-tests)

[Simulation tests](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html#simulation-tests)

[End-to-end tests](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html#end-to-end-tests)

#### **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)