[](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/7-integration-tests.html)



Search

[Interchain Developer Academy](https://ida.interchain.io/)[Interchain Developer Academy](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

Search



Filters

Interchain Developer Academy

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Week 0 - Getting Started](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Getting Started](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Blockchain 101](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Blockchain History](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Public and Managed Blockchains](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Consensus in Distributed Networks](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Cryptography](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Self-Assessment Quiz](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Go Introduction - First Steps](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Go Basics](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Go Interfaces](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Control Structures in Go](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Arrays and Slices in Go](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Standard Packages in Go](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Concurrency in Go](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Good-To-Know Dev Terms](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Docker Introduction](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Week 1 - Introduction to the Interchain](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Introduction to the Interchain](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Blockchain Technology and the Interchain](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[The Interchain Ecosystem](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Getting ATOM and Staking It](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[A Blockchain App Architecture](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Accounts](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Transactions](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Messages](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Modules](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Protobuf](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Multistore and Keepers](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[BaseApp](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Queries](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Events](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Context](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Testing](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Relaying with IBC](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Interchain Security](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Bridges](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Migrations](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Week 1 Quiz](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Week 2 - First Steps](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[First Steps](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Setup Your Work Environment](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Run a Node, API, and CLI](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Ignite CLI](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Exercise - Make a Checkers Blockchain](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Store Object](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Create Custom Messages](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Create and Save a Game Properly](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Add a Way to Make a Move](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Emit Game Information](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Record the Game Winner](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Week 2 Exercise](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Week 3 - Introduction to IBC and CosmJS](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Introduction to IBC and CosmJS](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[What is IBC?](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[IBC/TAO - Connections (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[IBC/TAO - Channels (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[IBC/TAO - Clients (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[IBC Token Transfer](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Interchain Accounts (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[IBC Middleware (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Create a Custom IBC Middleware (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Integrate IBC Middleware Into a Chain (OPTIONAL)](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[IBC Tooling](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[What is CosmJS?](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Your First CosmJS Actions](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Compose Complex Transactions](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Learn to Integrate Keplr](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Create Custom CosmJS Interfaces](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Week 4 - Ignite CLI and IBC Advanced](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Ignite CLI and IBC Advanced](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Keep an Up-To-Date Game Deadline](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Keep Track Of How Many Moves Have Been Played](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Put Your Games in Order](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Auto-Expiring Games](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Let Players Set a Wager](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Handle wager payments](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Integration tests](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Incentivize Players](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Help Find a Correct Move](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Play With Cross-Chain Tokens](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Understand IBC Denoms](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Go Relayer](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Hermes Relayer](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Week 5 - CosmJS Advanced](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[CosmJS Advanced](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Create Custom Objects](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Create Custom Messages](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Get an External GUI](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Integrate CosmJS and Keplr](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Backend Script for Game Indexing](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Week 6 - IBC Deep Dive](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[IBC Deep Dive](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[IBC Application Developer Introduction](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Make a Module IBC-Enabled](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Adding Packet and Acknowledgment Data](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Extend the Checkers Game With a Leaderboard](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Create a Leaderboard Chain](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

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

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

[Run in Production](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Prepare the Software to Run](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Prepare a Validator and Keys](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Prepare Where the Node Starts](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Prepare and Connect to Other Nodes](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Configure, Run, and Set Up a Service](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Prepare and Do Migrations](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Simulate Production in Docker](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Tally Player Info After Production](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Add a Leaderboard as a Module](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Migrate the Leaderboard Module After Production](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Simulate a Migration in Docker](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Final Exam](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[What's Next?](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

[Continue Your Interchain Journey](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html)

Docs Version Switcher

On this page

[What is to be done](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#what-is-to-be-done)

[Accommodate your code](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#accommodate-your-code)

[Test the test suite](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#test-the-test-suite)

[Helpers for money checking](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#helpers-for-money-checking)

[Anatomy of an integration suite test](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#anatomy-of-an-integration-suite-test)

[Extra tests](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#extra-tests)

[What happened to the events?](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#what-happened-to-the-events)

[Debug your suite](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#debug-your-suite)

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#integration-tests) **Integration Tests**



Make sure you have everything you need before proceeding:

* You understand the concepts of [modules](https://ida.interchain.io/academy/2-cosmos-concepts/5-modules.html), [keepers](https://ida.interchain.io/academy/2-cosmos-concepts/7-multistore-keepers.html), and [testing](https://ida.interchain.io/academy/2-cosmos-concepts/12-testing.html).
* Go is installed.
* You have the checkers blockchain codebase up to the wager payments. If not, follow the [previous steps](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/6-payment-winning.html) or check out [the relevant version (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/tree/payment-winning).



In this section, you will:

* Add integration tests.

In the [previous section](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/6-payment-winning.html), you handled wager payments and added unit tests that pass. You added mocks of the bank keeper. Mocks are useful to embody your expectations of the bank keeper's behavior and then quickly confirm that your code does what you expect. By interacting via the CLI, you also confirmed that the bank keeper behaved as you expected.

Now, it would be better to automatically check that your expectations of the bank keeper's behavior are correct. This is done with integration tests, and is the focus of this section.

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#what-is-to-be-done) What is to be done

In order, you will:

* Prepare your code to accept integration tests.
* Create helper functions that will make your integration tests more succinct.
* Add integration tests that create a full app and test proper token bank balances.



As a reminder:

* At version 0.45.4 of the Cosmos SDK, an integration test creates a full app.
* At version 0.47 of the SDK, an integration test creates a minimal app, and a test that creates a full app is called an end-to-end test (E2E).

Fortunately, you do not have to do this from scratch: taking inspiration from [tests on the bank module (opens new window)↗](https://github.com/cosmos/cosmos-sdk/blob/v0.45.4/x/bank/keeper/keeper_test.go#L66-L110), prepare your code so as to accommodate and create a full app that will contain a bank keeper, and add new tests.

For unit tests, each function takes a [t \*testing.T (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/x/checkers/keeper/end_block_server_game_test.go#L12) object. For integration tests, each function will be a method on a test suite that inherits from [testify's suite (opens new window)↗](https://pkg.go.dev/github.com/stretchr/testify/suite). This has the advantage that your test suite can have as many fields as is necessary or useful. The objects that you have used and would welcome in the suite are:



Copy

keeper keeper.Keeper

msgServer types.MsgServer

ctx sdk.Context

You can spread the suite's methods to different files, so as to keep consistent naming for your test files.

When testing, go test will find the suite because you add a [*regular* test (opens new window)↗](https://github.com/cosmos/cosmos-sdk/blob/v0.45.4/x/bank/keeper/keeper_test.go#L1233-L1235) that initializes the suite and runs it. The test suite is then automatically initialized with its [SetupTest (opens new window)↗](https://github.com/cosmos/cosmos-sdk/blob/v0.45.4/x/bank/keeper/keeper_test.go#L96) function via its parent suite class. After that, all the methods of the test suite are run.

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#accommodate-your-code) Accommodate your code

Copy and adjust from the Cosmos SDK.

1

Ignite CLI created a default constructor for your App with a [cosmoscmd.App (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/app/app.go#L245-L256) return type, but this is not convenient. Instead of risking breaking other dependencies, add a new constructor with [your App (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/app/app.go#L257-L281) as the return type.

2

Use [encoding.go (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/app/encoding.go) taken from [here (opens new window)↗](https://github.com/cosmos/cosmos-sdk/blob/v0.45.4/simapp/encoding.go), where you:

* Import "github.com/ignite-hq/cli/ignite/pkg/cosmoscmd".
* Replace simappparams.EncodingConfig with cosmoscmd.EncodingConfig.
* Replace simappparams.MakeTestEncodingConfig with appparams.MakeTestEncodingConfig.

3

Use [proto.go (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/app/params/proto.go) taken from [here (opens new window)↗](https://github.com/cosmos/cosmos-sdk/blob/v0.45.4/simapp/params/proto.go), where you:

* Import "github.com/ignite-hq/cli/ignite/pkg/cosmoscmd".
* Replace EncodingConfig with cosmoscmd.EncodingConfig.

4

Use [test\_helpers.go (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/app/test_helpers.go) taken from [here (opens new window)↗](https://github.com/cosmos/cosmos-sdk/blob/v0.45.4/simapp/test_helpers.go), in which you:

* Adjust from SimApp to App
* Adjust from New() to NewApp()
* Initialize your checkers genesis:



Copy

checkersGenesis := types.DefaultGenesis()

genesisState[types.ModuleName] = app.AppCodec().MustMarshalJSON(checkersGenesis)

app /

test\_helpers.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/app/test_helpers.go" \l "L146-L147" \t "_blank)

5

Define your test suite in a new keeper\_integration\_suite\_test.go file in a dedicated folder tests/integration/checkers/keeper:



Copy

type IntegrationTestSuite struct {

suite.Suite

app \*checkersapp.App

msgServer types.MsgServer

ctx sdk.Context

queryClient types.QueryClient

}

tests /

integration /

... /

keeper /

keeper\_integration\_suite\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/keeper_integration_suite_test.go" \l "L30-L37" \t "_blank)

6

Direct go test to it:



Copy

func TestCheckersKeeperTestSuite(t \*testing.T) {

suite.Run(t, new(IntegrationTestSuite))

}

tests /

integration /

... /

keeper /

keeper\_integration\_suite\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/keeper_integration_suite_test.go" \l "L43-L45" \t "_blank)

7

Create the suite.SetupTest function, taking inspiration from the [bank tests (opens new window)↗](https://github.com/cosmos/cosmos-sdk/blob/9e1ec7b/x/bank/keeper/keeper_test.go#L96-L110):



Copy

func (suite \*IntegrationTestSuite) SetupTest() {

app := checkersapp.Setup(false)

ctx := app.BaseApp.NewContext(false, tmproto.Header{Time: time.Now()})

app.AccountKeeper.SetParams(ctx, authtypes.DefaultParams())

app.BankKeeper.SetParams(ctx, banktypes.DefaultParams())

checkersModuleAddress = app.AccountKeeper.GetModuleAddress(types.ModuleName).String()

queryHelper := baseapp.NewQueryServerTestHelper(ctx, app.InterfaceRegistry())

types.RegisterQueryServer(queryHelper, app.CheckersKeeper)

queryClient := types.NewQueryClient(queryHelper)

suite.app = app

suite.msgServer = keeper.NewMsgServerImpl(app.CheckersKeeper)

suite.ctx = ctx

suite.queryClient = queryClient

}

tests /

integration /

... /

keeper /

keeper\_integration\_suite\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/keeper_integration_suite_test.go" \l "L47-L63" \t "_blank)

This [SetupTest function (opens new window)↗](https://github.com/stretchr/testify/blob/v1.7.0/suite/interfaces.go#L18-L22) is like a beforeEach as found in other test libraries. With it, you always get a new app in each test, without interference between them. Do not [omit it (opens new window)↗](https://github.com/stretchr/testify/blob/v1.7.0/suite/suite.go#L147) unless you have specific reasons to do so.



It collects your checkersModuleAddress for later use in tests that check events and balances:



Copy

var (

checkersModuleAddress string

)

tests /

integration /

... /

keeper /

keeper\_integration\_suite\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/keeper_integration_suite_test.go" \l "L39-L41" \t "_blank)

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#test-the-test-suite) Test the test suite

You can now confirm you did all this correctly by running these new keeper integration tests, although the suite has no tests. Note how the path to call has changed:

**Local**

**Docker**



Copy

$ go test github.com/alice/checkers/tests/integration/checkers/keeper

Copy

$ docker run --rm -it \

-v $(pwd):/checkers \

-w /checkers \

checkers\_i \

go test github.com/alice/checkers/tests/integration/checkers/keeper

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#helpers-for-money-checking) Helpers for money checking

Your upcoming integration tests will include checks on wagers being paid, lost, and won, so your tests need to initialize some bank balances for your players. This is made easier with a few helpers, including a helper to confirm a bank balance.

1. Make a bank genesis [Balance (opens new window)↗](https://github.com/cosmos/cosmos-sdk/blob/9e1ec7b6/x/bank/types/genesis.pb.go#L105-L110) type from primitives:



Copy

func makeBalance(address string, balance int64) banktypes.Balance {

return banktypes.Balance{

Address: address,

Coins: sdk.Coins{

sdk.Coin{

Denom: sdk.DefaultBondDenom,

Amount: sdk.NewInt(balance),

},

},

}

}

tests /

integration /

... /

keeper /

keeper\_integration\_suite\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/keeper_integration_suite_test.go" \l "L65-L75" \t "_blank)

1. Declare default accounts and balances that will be useful for you:



Copy

import (

"github.com/alice/checkers/x/checkers/testutil"

)

const (

alice = testutil.Alice

bob = testutil.Bob

carol = testutil.Carol

)

const (

balAlice = 50000000

balBob = 20000000

balCarol = 10000000

)

tests /

integration /

... /

keeper /

keeper\_integration\_suite\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/keeper_integration_suite_test.go" \l "L19-L28" \t "_blank)

1. Make your preferred bank genesis state:



Copy

func getBankGenesis() \*banktypes.GenesisState {

coins := []banktypes.Balance{

makeBalance(alice, balAlice),

makeBalance(bob, balBob),

makeBalance(carol, balCarol),

}

supply := banktypes.Supply{

Total: coins[0].Coins.Add(coins[1].Coins...).Add(coins[2].Coins...)

}

state := banktypes.NewGenesisState(

banktypes.DefaultParams(),

coins,

supply.GetTotal(),

[]banktypes.Metadata{})

return state

}

tests /

integration /

... /

keeper /

keeper\_integration\_suite\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/keeper_integration_suite_test.go" \l "L77-L94" \t "_blank)

1. Add a simple function to prepare your suite with your desired balances:



Copy

func (suite \*IntegrationTestSuite) setupSuiteWithBalances() {

suite.app.BankKeeper.InitGenesis(suite.ctx, getBankGenesis())

}

tests /

integration /

... /

keeper /

keeper\_integration\_suite\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/keeper_integration_suite_test.go" \l "L96-L98" \t "_blank)

1. Add a function to check balances from primitives:



Copy

func (suite \*IntegrationTestSuite) RequireBankBalance(expected int, atAddress string) {

sdkAdd, err := sdk.AccAddressFromBech32(atAddress)

suite.Require().Nil(err, "Failed to parse address: %s", atAddress)

suite.Require().Equal(

int64(expected),

suite.app.BankKeeper.GetBalance(suite.ctx, sdkAdd, sdk.DefaultBondDenom).Amount.Int64())

}

tests /

integration /

... /

keeper /

keeper\_integration\_suite\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/keeper_integration_suite_test.go" \l "L100-L106" \t "_blank)

With the preparation done, what does an integration test method look like?

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#anatomy-of-an-integration-suite-test) Anatomy of an integration suite test

Now you must add integration tests for your keeper in new files. What does an integration test look like? Take the example of a [simple unit test (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/x/checkers/keeper/msg_server_create_game_test.go#L35-L66) ported to the integration test suite:

1. The method has a declaration:



Copy

func (suite \*IntegrationTestSuite) TestCreate1GameHasSaved()

tests /

integration /

... /

keeper /

msg\_server\_create\_game\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/msg_server_create_game_test.go" \l "L8" \t "_blank)

It is declared as a member of your test suite, and is prefixed with [Test (opens new window)↗](https://github.com/stretchr/testify/blob/v1.7.0/suite/suite.go#L181-L182).

1. The **setup** can be done as you prefer, but just like for unit tests you ought to create a helper and use it. Here one exists already:



Copy

suite.setupSuiteWithBalances()

goCtx := sdk.WrapSDKContext(suite.ctx)

tests /

integration /

... /

keeper /

msg\_server\_create\_game\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/msg_server_create_game_test.go" \l "L9-L10" \t "_blank)

1. The **action** is no different from a unit test's action, other than that you get the keeper or msgServer from the suite's fields:



Copy

suite.msgServer.CreateGame(goCtx, &types.MsgCreateGame{

Creator: alice,

Red: bob,

Black: carol,

Wager: 45,

})

keeper := suite.app.CheckersKeeper

tests /

integration /

... /

keeper /

msg\_server\_create\_game\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/msg_server_create_game_test.go" \l "L11-L17" \t "_blank)

1. The **verification** is done with suite.Require().X, but otherwise looks similar to the shorter require.X of unit tests:



Copy

systemInfo, found := keeper.GetSystemInfo(suite.ctx)

suite.Require().True(found)

suite.Require().EqualValues(types.SystemInfo{

NextId: 2,

FifoHeadIndex: "1",

FifoTailIndex: "1",

}, systemInfo)

tests /

integration /

... /

keeper /

msg\_server\_create\_game\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/msg_server_create_game_test.go" \l "L18-L24" \t "_blank)

In fact, it is exactly the [same require (opens new window)↗](https://github.com/stretchr/testify/blob/v1.7.0/suite/suite.go#L24) object.

You have added an integration test that copies an existing unit test. It demonstrates the concept but is of limited additional utility.

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#extra-tests) Extra tests

It is time to add extra tests that check money handling by the bank. Before jumping in, as you did in *play* unit tests you can add a method that prepares your suite's keeper with a game ready to be played on:



Copy

func (suite \*IntegrationTestSuite) setupSuiteWithOneGameForPlayMove() {

suite.setupSuiteWithBalances()

goCtx := sdk.WrapSDKContext(suite.ctx)

suite.msgServer.CreateGame(goCtx, &types.MsgCreateGame{

Creator: alice,

Red: bob,

Black: carol,

Wager: 45,

})

}

tests /

integration /

... /

keeper /

msg\_server\_play\_move\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/msg_server_play_move_test.go" \l "L9-L18" \t "_blank)

You will call this function from the relevant tests.

For the tests proper, before an action that you expect to transfer money (or not), you can verify the initial position:



Copy

suite.RequireBankBalance(balAlice, alice)

suite.RequireBankBalance(balBob, bob)

suite.RequireBankBalance(balCarol, carol)

suite.RequireBankBalance(0, checkersModuleAddress)

tests /

integration /

... /

keeper /

msg\_server\_play\_move\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/msg_server_play_move_test.go" \l "L59-L62" \t "_blank)

After the action you can test the new balances, for instance:



Copy

suite.RequireBankBalance(balAlice, alice)

suite.RequireBankBalance(balBob-45, bob)

suite.RequireBankBalance(balCarol, carol)

suite.RequireBankBalance(45, checkersModuleAddress)

tests /

integration /

... /

keeper /

msg\_server\_play\_move\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/msg_server_play_move_test.go" \l "L71-L74" \t "_blank)

How you subdivide your tests and where you insert these balance checks is up to you. You can find examples here for:

* [Creating a game (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/msg_server_create_game_test.go#L42-L59).
* [Playing the first move (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/msg_server_play_move_test.go#L56-L75) and [the second move (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/msg_server_play_move_test.go#L209-L236), including [up to a resolution (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/msg_server_play_move_test.go#L308-L315). You can also [check the events (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/msg_server_play_move_test.go#L129-L164).
* Failing to play a game because of a failure to pay the wager on the [first move (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/msg_server_play_move_test.go#L104-L127) and the [second move (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/msg_server_play_move_test.go#L238-L269).
* [Forfeiting a game (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/end_block_server_game_test.go#L10-L30), including when [there has been one move played (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/end_block_server_game_test.go#L32-L60) or [two (opens new window)↗](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/end_block_server_game_test.go#L185-L222).

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#what-happened-to-the-events) What happened to the events?

With the new tests, you may think that the events are compromised. For instance, the event type "transfer" normally comes with three attributes, but when the bank has made two transfers the "transfer" event ends up with 6 attributes. This is just the way events are organized: per type, with the attributes piled in.

When checking emitted events, you need to skip over the attributes you are not checking. You can easily achieve that with [Go slices](https://ida.interchain.io/tutorials/4-golang-intro/5-arrays.html).

For instance, here transferEvent.Attributes[6:] discards the first six attributes:



Copy

transferEvent := events[6]

suite.Require().Equal(transferEvent.Type, "transfer")

suite.Require().EqualValues([]sdk.Attribute{

{Key: "recipient", Value: carol},

{Key: "sender", Value: checkersModuleAddress},

{Key: "amount", Value: "90stake"},

}, transferEvent.Attributes[6:])

tests /

integration /

... /

keeper /

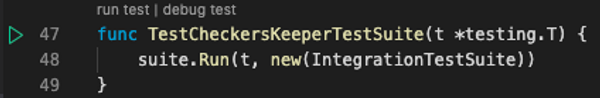
end\_block\_server\_game\_test.go

[View source→](https://github.com/cosmos/b9-checkers-academy-draft/blob/integration-tests/tests/integration/checkers/keeper/end_block_server_game_test.go" \l "L264-L270" \t "_blank)

[#Copy link](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#debug-your-suite) Debug your suite

You learned in a [previous section](https://ida.interchain.io/hands-on-exercise/1-ignite-cli/3-stored-game.html) how to launch a test in debug mode. It is still possible to do so when using a suite. Depending on the versions of your Go installation and your Visual Studio Code, you can launch it in two ways:

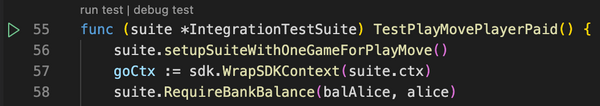
1. Right-click on the arrow to the left of the suite's runner func TestCheckersKeeperTestSuite:





In this case, you can only launch debug for **all** of the suite's test methods and not just a single one (as is possible with a simple test).

1. Right-click on the arrow to the left of the separate test of the suite:





This option may not be available. If being able to debug only a few tests at a time is important to you, a solution is to create more granular suites, for example using one or more test suites per file and falling back on the first option.

With this you have tested your wager payments in a way more realistic that unit tests and mocks.

synopsis

To summarize, this section has explored:

* How to prepare your code so as to accommodate integration tests.
* How an integration test is built, and what is a test suite.
* How to add simple integration tests and helpers.
* How to add meaningful integration tests and account for how the events are emitted.
* How to debug your integration tests.

previous

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/6-payment-winning.html)

**[Handle wager payments](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/6-payment-winning.html)**

up next

**[Incentivize Players](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/8-gas-meter.html)**

[[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/8-gas-meter.html)](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/8-gas-meter.html)

Rate this Page

icon smile

icon meh

icon frown

Would you like to add a message?

Submit

Thank you for your Feedback!

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

On this page

[What is to be done](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#what-is-to-be-done)

[Accommodate your code](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#accommodate-your-code)

[Test the test suite](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#test-the-test-suite)

[Helpers for money checking](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#helpers-for-money-checking)

[Anatomy of an integration suite test](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#anatomy-of-an-integration-suite-test)

[Extra tests](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#extra-tests)

[What happened to the events?](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#what-happened-to-the-events)

[Debug your suite](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/7-integration-tests.html#debug-your-suite)

#### **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)

Original text

Rate this translation

Your feedback will be used to help improve Google Translate