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

[Interchain Developer Academy](https://ida.interchain.io/)/[Interchain Developer Academy](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)



Search

[Interchain Developer Academy](https://ida.interchain.io/)[Interchain Developer Academy](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

Search



Filters

Interchain Developer Academy

[](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Week 0 - Getting Started](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Getting Started](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Blockchain 101](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Blockchain History](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Public and Managed Blockchains](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Consensus in Distributed Networks](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Cryptography](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Self-Assessment Quiz](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Go Introduction - First Steps](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Go Basics](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Go Interfaces](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Control Structures in Go](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Arrays and Slices in Go](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Standard Packages in Go](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Concurrency in Go](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Good-To-Know Dev Terms](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Docker Introduction](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Week 1 - Introduction to the Interchain](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Introduction to the Interchain](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Blockchain Technology and the Interchain](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[The Interchain Ecosystem](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Getting ATOM and Staking It](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[A Blockchain App Architecture](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Accounts](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Transactions](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Messages](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Modules](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Protobuf](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Multistore and Keepers](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[BaseApp](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Queries](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Events](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Context](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Testing](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Relaying with IBC](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Interchain Security](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Bridges](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Migrations](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Week 1 Quiz](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Week 2 - First Steps](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[First Steps](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Setup Your Work Environment](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Run a Node, API, and CLI](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Ignite CLI](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Exercise - Make a Checkers Blockchain](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Store Object](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Create Custom Messages](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Create and Save a Game Properly](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Add a Way to Make a Move](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Emit Game Information](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Record the Game Winner](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Week 2 Exercise](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Week 3 - Introduction to IBC and CosmJS](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Introduction to IBC and CosmJS](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[What is IBC?](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[IBC/TAO - Connections (OPTIONAL)](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[IBC/TAO - Channels (OPTIONAL)](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[IBC/TAO - Clients (OPTIONAL)](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[IBC Token Transfer](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Interchain Accounts (OPTIONAL)](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[IBC Middleware (OPTIONAL)](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Create a Custom IBC Middleware (OPTIONAL)](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Integrate IBC Middleware Into a Chain (OPTIONAL)](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[IBC Tooling](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[What is CosmJS?](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Your First CosmJS Actions](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Compose Complex Transactions](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Learn to Integrate Keplr](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Create Custom CosmJS Interfaces](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Week 4 - Ignite CLI and IBC Advanced](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Ignite CLI and IBC Advanced](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Keep an Up-To-Date Game Deadline](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Keep Track Of How Many Moves Have Been Played](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Put Your Games in Order](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Auto-Expiring Games](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Let Players Set a Wager](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Handle wager payments](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Integration tests](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Incentivize Players](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Help Find a Correct Move](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Play With Cross-Chain Tokens](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Understand IBC Denoms](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Go Relayer](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Hermes Relayer](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Week 5 - CosmJS Advanced](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[CosmJS Advanced](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Create Custom Objects](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Create Custom Messages](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Get an External GUI](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Integrate CosmJS and Keplr](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Backend Script for Game Indexing](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Week 6 - IBC Deep Dive](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[IBC Deep Dive](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[IBC Application Developer Introduction](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Make a Module IBC-Enabled](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Adding Packet and Acknowledgment Data](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Extend the Checkers Game With a Leaderboard](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Create a Leaderboard Chain](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Week 7 - From Code to MVP to Production and Migrations](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[From Code to MVP to Production and Migrations](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Run in Production](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Prepare the Software to Run](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Prepare a Validator and Keys](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Prepare Where the Node Starts](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Prepare and Connect to Other Nodes](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Configure, Run, and Set Up a Service](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Prepare and Do Migrations](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Simulate Production in Docker](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Tally Player Info After Production](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Add a Leaderboard as a Module](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Migrate the Leaderboard Module After Production](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Simulate a Migration in Docker](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Final Exam](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[What's Next?](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

[Continue Your Interchain Journey](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html)

Docs Version Switcher

On this page

[Compiling the Protobuf objects and clients](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#compiling-the-protobuf-objects-and-clients)

[Preparation](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#preparation)

[Getting third party files](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#getting-third-party-files)

[Compilation](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#compilation)

[A note about the result](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#a-note-about-the-result)

[Proper saving](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#proper-saving)

[Add convenience with types](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#add-convenience-with-types)

[For messages](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#for-messages)

[For queries](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#for-queries)

[Integration with Stargate](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#integration-with-stargate)

[#Copy link](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#create-custom-cosmjs-interfaces) **Create Custom CosmJS Interfaces**



In this section, you will:

* Create custom CosmJS interfaces to connect to custom Cosmos SDK modules.
* Define custom interfaces with Protobuf.
* Define custom types and messages.
* Integrate with Ignite - previously known as Starport.

CosmJS comes out of the box with interfaces that connect with the standard Cosmos SDK modules such as bank and gov and understand the way their messages are serialized. Since your own blockchain's modules are unique, they need custom CosmJS interfaces. That process consists of several steps:

1. Creating the Protobuf objects and clients in TypeScript.
2. Creating extensions that facilitate the use of the above clients.
3. Any further level of abstraction that you deem useful for integration.

This section assumes that you have a working Cosmos blockchain with its own modules. It is based on CosmJS version [v0.28.3 (opens new window)↗](https://github.com/cosmos/cosmjs/tree/v0.28.3).

[#Copy link](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#compiling-the-protobuf-objects-and-clients) Compiling the Protobuf objects and clients

You can choose which library you use to compile your Protobuf objects into TypeScript or JavaScript. Reproducing [what Stargate (opens new window)↗](https://github.com/cosmos/cosmjs/blob/main/packages/stargate/CUSTOM_PROTOBUF_CODECS.md) or [cosmjs-types (opens new window)↗](https://github.com/confio/cosmjs-types/blob/main/scripts/codegen.js) do is a good choice.

[#Copy link](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#preparation) Preparation

This exercise assumes that:

1. Your Protobuf definition files are in ./proto/myChain.
2. You want to compile them into TypeScript in ./client/src/types/generated.

Install protoc on your computer and its Typescript plugin in your project, possibly with the help of a Dockerfile:

**Local**

**Dockerfile**



Copy

$ mkdir -p /usr/lib/protoc

$ cd /usr/lib/protoc

$ curl -L https://github.com/protocolbuffers/protobuf/releases/download/v21.7/protoc-21.7-linux-x86\_64.zip -o protoc.zip

$ unzip -o protoc.zip

$ rm protoc.zip

$ ln -s /usr/lib/protoc/bin/protoc /usr/local/bin/protoc

$ npm install ts-proto@1.121.6 --save-dev

Adjust to your preferred version, operating system, and CPU platform. For instance, on an Apple M1 you would use protoc-21.7-osx-aarch\_64.zip.

Copy

FROM --platform=linux node:lts-slim as base

ARG BUILDARCH

ENV PROTOC\_VERSION=21.7

ENV TS\_PROTO\_VERSION=1.121.6

FROM base AS platform-amd64

ENV PROTOC\_PLATFORM=x86\_64

FROM base AS platform-arm64

ENV PROTOC\_PLATFORM=aarch\_64

FROM platform-${BUILDARCH}

RUN apt-get update

RUN apt-get install curl unzip --yes

# Install ProtoC

RUN mkdir -p /usr/lib/protoc

WORKDIR /usr/lib/protoc

RUN curl -L https://github.com/protocolbuffers/protobuf/releases/download/v${PROTOC\_VERSION}/protoc-${PROTOC\_VERSION}-linux-${PROTOC\_PLATFORM}.zip -o protoc.zip

RUN unzip -o protoc.zip

RUN rm protoc.zip

RUN ln -s /usr/lib/protoc/bin/protoc /usr/local/bin/protoc

# Install ts-proto

RUN npm install --global ts-proto@${TS\_PROTO\_VERSION} --save-exact

WORKDIR /

ENTRYPOINT [ "protoc" ]

Then build the image:

Copy

$ docker build . -t ts-protoc

You can confirm the version you received. The executable is located in ./node\_modules/protoc/protoc/bin/protoc:

**Local**

**Docker**



Copy

$ protoc --version

Copy

$ docker run --rm -it \

ts-protoc --version

This returns something like:



Copy

libprotoc 3.21.7

The compiler tools are ready. Time to use them.

Create the target folder if it does not exist yet:



Copy

$ mkdir -p client/src/types/generated

[#Copy link](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#getting-third-party-files) Getting third party files

You need to get the imports that appear in your .proto files. Usually you can find the following in [query.proto (opens new window)↗](https://github.com/cosmos/cosmos-sdk/blob/d98503b/proto/cosmos/bank/v1beta1/query.proto#L4-L6):



Copy

import "cosmos/base/query/v1beta1/pagination.proto";

import "gogoproto/gogo.proto";

import "google/api/annotations.proto";

You need local copies of the right file versions in the right locations. Pay particular attention to Cosmos SDK's version of your project. You can check by running:



Copy

$ grep cosmos-sdk go.mod

This returns something like:



Copy

github.com/cosmos/cosmos-sdk v0.45.4

Use this version as a tag on Github. One way to retrieve the [pagination file (opens new window)↗](https://github.com/cosmos/cosmos-sdk/blob/v0.45.4/proto/cosmos/base/query/v1beta1/pagination.proto) is:



Copy

$ mkdir -p ./proto/cosmos/base/query/v1beta1/

$ curl https://raw.githubusercontent.com/cosmos/cosmos-sdk/v0.45.4/proto/cosmos/base/query/v1beta1/pagination.proto -o ./proto/cosmos/base/query/v1beta1/pagination.proto

You can do the same for the others, found in the [third\_party folder (opens new window)↗](https://github.com/cosmos/cosmos-sdk/tree/v0.45.4/third_party/proto) under the same version:



Copy

$ mkdir -p ./proto/google/api

$ curl https://raw.githubusercontent.com/cosmos/cosmos-sdk/v0.45.4/third\_party/proto/google/api/annotations.proto -o ./proto/google/api/annotations.proto

$ curl https://raw.githubusercontent.com/cosmos/cosmos-sdk/v0.45.4/third\_party/proto/google/api/http.proto -o ./proto/google/api/http.proto

$ mkdir -p ./proto/gogoproto

$ curl https://raw.githubusercontent.com/cosmos/cosmos-sdk/v0.45.4/third\_party/proto/gogoproto/gogo.proto -o ./proto/gogoproto/gogo.proto

[#Copy link](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#compilation) Compilation

You can now compile the Protobuf files. To avoid adding all the .proto files manually to the command, use xargs:

**Local**

**Docker**



Copy

$ ls ./proto/myChain | xargs -I {} protoc \

--plugin="./node\_modules/ts-proto/protoc-gen-ts\_proto" \

--ts\_proto\_out="./client/src/types/generated" \

--proto\_path="./proto" \

--ts\_proto\_opt="esModuleInterop=true,forceLong=long,useOptionals=messages" \

myChain/{}

Copy

$ ls ./proto/myChain | xargs -I {} \

docker run --rm -i \

-v $(pwd):/project -w /project \

ts-protoc \

--plugin="/usr/local/lib/node\_modules/ts-proto/protoc-gen-ts\_proto" \

--ts\_proto\_out="./client/src/types/generated" \

--proto\_path="./proto" \

--ts\_proto\_opt="esModuleInterop=true,forceLong=long,useOptionals=messages" \

myChain/{}

Where /usr/local/lib/node\_modules is the result of the query:

Copy

$ docker run --rm -it \

--entrypoint npm \

ts-protoc \

root --global

This shows where ts-proto was installed globally.

--proto\_path is only ./proto so that your imports (such as import "cosmos/base...) can be found.

You should now see your files compiled into TypeScript. They have been correctly filed under their respective folders and contain both types and services definitions. It also created the compiled versions of your third party imports.

[#Copy link](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#a-note-about-the-result) A note about the result

Your tx.proto file may have contained the following:



Copy

service Msg {

rpc Send(MsgSend) returns (MsgSendResponse);

//...

}

proto /

cosmos /

... /

v1beta1 /

tx.proto

[View source→](https://github.com/cosmos/cosmos-sdk/blob/v0.45.4/proto/cosmos/bank/v1beta1/tx.proto" \l "L11-L17" \t "_blank)

If so, you find its service declaration in the compiled tx.ts file:



Copy

export interface Msg {

Send(request: MsgSend): Promise<MsgSendResponse>;

//...

}

src /

cosmos /

... /

v1beta1 /

tx.ts

[View source→](https://github.com/confio/cosmjs-types/blob/v0.4.1/src/cosmos/bank/v1beta1/tx.ts" \l "L243-L248" \t "_blank)

It also appears in the default implementation:



Copy

export class MsgClientImpl implements Msg {

private readonly rpc: Rpc;

constructor(rpc: Rpc) {

this.rpc = rpc;

this.Send = this.Send.bind(this);

//...

}

Send(request: MsgSend): Promise<MsgSendResponse> {

const data = MsgSend.encode(request).finish();

const promise = this.rpc.request("cosmos.bank.v1beta1.Msg", "Send", data);

return promise.then((data) => MsgSendResponse.decode(new \_m0.Reader(data)));

}

//...

}

src /

cosmos /

... /

v1beta1 /

tx.ts

[View source→](https://github.com/confio/cosmjs-types/blob/v0.4.1/src/cosmos/bank/v1beta1/tx.ts" \l "L250-L268" \t "_blank)

The important points to remember from this are:

1. rpc: RPC is an instance of a Protobuf RPC client that is given to you by CosmJS. Although the interface appears to be [declared locally (opens new window)↗](https://github.com/confio/cosmjs-types/blob/v0.4.1/src/cosmos/bank/v1beta1/tx.ts#L270-L272), this is the same interface found [throughout CosmJS (opens new window)↗](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/queryclient/utils.ts#L35-L37). It is given to you [on construction (opens new window)↗](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/queryclient/queryclient.ts). At this point you do not need an implementation for it.
2. You can see encode and decode in action. Notice the .finish() that flushes the Protobuf writer buffer.
3. The rpc.request makes calls that are correctly understood by the Protobuf compiled server on the other side.

You can find the same structure in [query.ts (opens new window)↗](https://github.com/confio/cosmjs-types/blob/v0.4.1/src/cosmos/bank/v1beta1/query.ts).

[#Copy link](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#proper-saving) Proper saving

Commit the extra .proto files as well as the compiled ones to your repository so you do not need to recreate them.

Take inspiration from cosmjs-types [codegen.sh (opens new window)↗](https://github.com/confio/cosmjs-types/tree/main/scripts):

1. Create a script file named ts-proto.sh with the previous command, or create a Makefile target.
2. Add an [npm run target (opens new window)↗](https://github.com/confio/cosmjs-types/blob/c64759a/package.json#L31) with it, to keep track of how this was done and easily reproduce it in the future when you update a Protobuf file.

[#Copy link](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#add-convenience-with-types) Add convenience with types

CosmJS provides an interface to which all the created types conform, [TsProtoGeneratedType (opens new window)↗](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/proto-signing/src/registry.ts#L12-L18), which is itself a sub-type of [GeneratedType (opens new window)↗](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/proto-signing/src/registry.ts#L32). In the same file, note the definition:



Copy

export interface EncodeObject {

readonly typeUrl: string;

readonly value: any;

}

packages /

proto-signing /

src /

registry.ts

[View source→](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/proto-signing/src/registry.ts" \l "L54-L57" \t "_blank)

The typeUrl is the identifier by which Protobuf identifies the type of the data to serialize or deserialize. It is composed of the type's package and its name. For instance (and see also [here (opens new window)↗](https://github.com/cosmos/cosmos-sdk/blob/3a1027c/proto/cosmos/bank/v1beta1/tx.proto)):



Copy

package cosmos.bank.v1beta1;

//...

message MsgSend {

//...

}

In this case, the MsgSend's type URL is ["/cosmos.bank.v1beta1.MsgSend" (opens new window)↗](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/modules/bank/messages.ts#L6).

Each of your types is associated like this. You can declare each string as a constant value, such as:



Copy

export const msgSendTypeUrl = "/cosmos.bank.v1beta1.MsgSend";

Save those along with generated in ./client/src/types/modules.

[#Copy link](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#for-messages) For messages

Messages, sub-types of Msg, are assembled into transactions that are then sent to CometBFT. CosmJS types already include types for [transactions (opens new window)↗](https://github.com/confio/cosmjs-types/blob/v0.4.1/src/cosmos/tx/v1beta1/tx.ts#L12-L26). These are assembled, signed, and sent by the [SigningStargateClient (opens new window)↗](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/signingstargateclient.ts#L280-L298) of CosmJS.

The Msg kind also needs to be added to a registry. To facilitate that, you should prepare them in a nested array:



Copy

export const bankTypes: ReadonlyArray<[string, GeneratedType]> = [

["/cosmos.bank.v1beta1.MsgMultiSend", MsgMultiSend],

["/cosmos.bank.v1beta1.MsgSend", MsgSend],

];

packages /

stargate /

... /

bank /

messages.ts

[View source→](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/modules/bank/messages.ts" \l "L4-L7" \t "_blank)

Add child types to EncodeObject to direct Typescript:



Copy

export interface MsgSendEncodeObject extends EncodeObject {

readonly typeUrl: "/cosmos.bank.v1beta1.MsgSend";

readonly value: Partial<MsgSend>;

}

packages /

stargate /

... /

bank /

messages.ts

[View source→](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/modules/bank/messages.ts" \l "L9-L12" \t "_blank)

In the previous code, you cannot reuse your msgSendTypeUrl because it is a value not a type. You can add a type helper, which is useful in an if else situation:



Copy

export function isMsgSendEncodeObject(encodeObject: EncodeObject): encodeObject is MsgSendEncodeObject {

return (encodeObject as MsgSendEncodeObject).typeUrl === "/cosmos.bank.v1beta1.MsgSend";

}

packages /

stargate /

... /

bank /

messages.ts

[View source→](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/modules/bank/messages.ts" \l "L14-L16" \t "_blank)

[#Copy link](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#for-queries) For queries

Queries have very different types of calls. It makes sense to organize them in one place, called an extension. For example:



Copy

export interface BankExtension {

readonly bank: {

readonly balance: (address: string, denom: string) => Promise<Coin>;

readonly allBalances: (address: string) => Promise<Coin[]>;

//...

};

}

packages /

stargate /

... /

bank /

queries.ts

[View source→](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/modules/bank/queries.ts" \l "L9-L18" \t "_blank)

Note that there is a **key** bank: inside it. This becomes important later on when you *add* it to Stargate.

1. Create an extension interface for your module using function names and parameters that satisfy your needs.
2. It is recommended to make sure that the key is unique and does not overlap with any other modules of your application.
3. Create a factory for its implementation copying the [model here (opens new window)↗](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/modules/bank/queries.ts#L20-L59). Remember that the [QueryClientImpl (opens new window)↗](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/modules/bank/queries.ts#L4) implementation must come from your own compiled Protobuf query service.

[#Copy link](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#integration-with-stargate) Integration with Stargate

StargateClient and SigningStargateClient are typically the ultimate abstractions that facilitate the querying and sending of transactions. You are now ready to add your own elements to them. The easiest way is to inherit from them and expose the extra functions you require.

If your extra functions map one-for-one with those of your own extension, then you can publicly expose the extension itself to minimize duplication in [StargateClient (opens new window)↗](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/stargateclient.ts#L143) and [SigningStargateClient (opens new window)↗](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/signingstargateclient.ts#L109).

For example, if you have your interface MyExtension with a myKey key and you are creating MyStargateClient:



Copy

export class MyStargateClient extends StargateClient {

public readonly myQueryClient: MyExtension | undefined

public static async connect(

endpoint: string,

options: StargateClientOptions = {},

): Promise<MyStargateClient> {

const tmClient = await Tendermint34Client.connect(endpoint)

return new MyStargateClient(tmClient, options)

}

protected constructor(tmClient: Tendermint34Client | undefined, options: StargateClientOptions) {

super(tmClient, options)

if (tmClient) {

this.myQueryClient = QueryClient.withExtensions(tmClient, setupMyExtension)

}

}

}

You can extend [StargateClientOptions (opens new window)↗](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/stargateclient.ts#L139-L141) if your own client can receive further options.

You also need to inform MySigningStargateClient about the extra encodable types it should be able to handle. The list is defined in a registry that you can [pass as options (opens new window)↗](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/signingstargateclient.ts#L139).

Take inspiration from the [SigningStargateClient source code (opens new window)↗](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/signingstargateclient.ts#L76-L80) itself. Collect your new types into an array:



Copy

import { defaultRegistryTypes } from "@cosmjs/stargate"

export const myDefaultRegistryTypes: ReadonlyArray<[string, GeneratedType]> = [

...defaultRegistryTypes,

...myTypes, // As you defined bankTypes earlier

]

Taking inspiration from [the same place (opens new window)↗](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/signingstargateclient.ts#L118-L120), add the registry creator:



Copy

function createDefaultRegistry(): Registry {

return new Registry(myDefaultRegistryTypes)

}

Now you are ready to combine this into your own MySigningStargateClient. It still takes an optional registry, but if that is missing it adds your newly defined default one:



Copy

export class MySigningStargateClient extends SigningStargateClient {

public readonly myQueryClient: MyExtension | undefined

public static async connectWithSigner(

endpoint: string,

signer: OfflineSigner,

options: SigningStargateClientOptions = {}

): Promise<MySigningStargateClient> {

const tmClient = await Tendermint34Client.connect(endpoint)

return new MySigningStargateClient(tmClient, signer, {

registry: createDefaultRegistry(),

...options,

})

}

protected constructor(tmClient: Tendermint34Client | undefined, signer: OfflineSigner, options: SigningStargateClientOptions) {

super(tmClient, signer, options)

if (tmClient) {

this.myQueryClient = QueryClient.withExtensions(tmClient, setupMyExtension)

}

}

}

You can optionally add dedicated functions that use your own types, modeled on:



Copy

public async sendTokens(

senderAddress: string,

recipientAddress: string,

amount: readonly Coin[],

fee: StdFee | "auto" | number,

memo = "",

): Promise<DeliverTxResponse> {

const sendMsg: MsgSendEncodeObject = {

typeUrl: "/cosmos.bank.v1beta1.MsgSend",

value: {

fromAddress: senderAddress,

toAddress: recipientAddress,

amount: [...amount],

},

};

return this.signAndBroadcast(senderAddress, [sendMsg], fee, memo);

}

packages /

stargate /

src /

signingstargateclient.ts

[View source→](https://github.com/cosmos/cosmjs/blob/v0.28.3/packages/stargate/src/signingstargateclient.ts" \l "L180-L196" \t "_blank)

Think of your functions as examples of proper use, that other developers can reuse when assembling more complex transactions.

You are ready to import and use this in a server script or a GUI.



If you would like to get started on building your own CosmJS elements on your own checkers game, you can go straight to the exercise in [CosmJS for Your Chain](https://ida.interchain.io/hands-on-exercise/3-cosmjs-adv/) to start from scratch.

More specifically, you can jump to:

* [Create Custom Objects](https://ida.interchain.io/hands-on-exercise/3-cosmjs-adv/1-cosmjs-objects.html), to see how to compile the Protobuf objects.
* [Create Custom Messages](https://ida.interchain.io/hands-on-exercise/3-cosmjs-adv/2-cosmjs-messages.html), to see how to create messages relevant for checkers.
* [Backend Script for Game Indexing](https://ida.interchain.io/hands-on-exercise/3-cosmjs-adv/5-server-side.html), to see how this can be used also to listen to events coming from the blockchain.
* [Integrate CosmJS and Keplr](https://ida.interchain.io/hands-on-exercise/3-cosmjs-adv/4-cosmjs-gui.html), to see how to use and integrate what you prepared into a preexisting Checkers GUI.

synopsis

To summarize, this section has explored:

* How CosmJS's out-of-the-box interfaces understand how messages of standard Cosmos SDK modules are serialized, meaning that your unique modules will require custom CosmJS interfaces of their own.
* How to create the necessary Protobuf objects and clients in Typescript, the extensions that facilitate the use of these clients, and any further level of abstraction that you deem useful for integration.
* How to integrate CosmJS with Ignite's client and signing client, which are typically the ultimate abstractions that facilitate the querying and sending of transactions.

previous

[](https://ida.interchain.io/tutorials/7-cosmjs/4-with-keplr.html)

**[Learn to Integrate Keplr](https://ida.interchain.io/tutorials/7-cosmjs/4-with-keplr.html)**

up next

**[Ignite CLI and IBC Advanced](https://ida.interchain.io/ida-course/LPs/week-4/)**

[[](https://ida.interchain.io/ida-course/LPs/week-4/)](https://ida.interchain.io/ida-course/LPs/week-4/)

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

[Compiling the Protobuf objects and clients](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#compiling-the-protobuf-objects-and-clients)

[Preparation](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#preparation)

[Getting third party files](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#getting-third-party-files)

[Compilation](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#compilation)

[A note about the result](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#a-note-about-the-result)

[Proper saving](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#proper-saving)

[Add convenience with types](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#add-convenience-with-types)

[For messages](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#for-messages)

[For queries](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#for-queries)

[Integration with Stargate](https://ida.interchain.io/tutorials/7-cosmjs/5-create-custom.html#integration-with-stargate)

#### **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)