# The Move Book

In the previous section we created a new package and demonstrated the basic flow of creating, building, and testing a Move package. In this section, we will write a simple application that uses the storage model and can be interacted with. To do this, we will create a simple todo list application.

Following the same flow as in Hello, World! , we will create a new package called todo_list .

To speed things up and focus on the application logic, we will provide the code for the todo list application. Replace the contents of the sources/todo_list.move file with the following code:

Note: while the contents may seem overwhelming at first, we will break it down in the following sections. Try to focus on what's at hand right now.

To make sure that we did everything correctly, let's build the package by running the sui move build command. If everything is correct, you should see the output similar to the following:

If there are no errors following this output, you have successfully built the package. If there are errors, make sure that:

There are not many other reasons for the code to fail at this stage. But if you are still having issues, try looking up the structure of the package in this location .

If you already have an account set up, you can skip this step.

To publish and interact with the package, we need to set up an account. While developing, the best option for doing so is to run your own Local Network . For now you just need to run RUST_LOG="off,sui_node=info" sui start --with-faucet --force-regenesis . The Sui Local Network will run on port 9000 of your machine, so make sure that the port isn't being used by any other application.

If you are doing it for the first time, you will need to create a new account. To do this, run the sui client command, then the CLI will prompt you with multiple questions. The answers are marked below with > :

After you have answered the questions, the CLI will generate a new keypair and save it to the configuration file. You can now use this account to interact with the network.

To check that we have the account set up correctly, run the sui client active-address command:

The command will output the address of your account, it starts with 0x followed by 64 characters.

In devnet and testnet environments, the CLI provides a way to request coins to your account, so you can interact with the network. To request coins, run the sui client faucet command:

After waiting a little bit, you can check that the Coin object was sent to your account by running the sui client balance command:

Alternatively, you can query objects owned by your account, by running the sui client objects command. The actual output will be different, because the object ID is unique, and so is digest, but the structure will be similar:

Now that we have the account set up and the coins in the account, we can interact with the network. We will start by publishing the package to the network.

To publish the package to the network, we will use the sui client publish command. The command will automatically build the package and use its bytecode to publish in a single transaction.

We are using the --gas-budget argument during publishing. It specifies how much gas we are willing to spend on the transaction. We won't touch on this topic in this section, but it's important to know that every transaction in Sui costs gas, and the gas is paid in SUI coins.

The gas-budget is specified in MISTs . 1 SUI equals 10^9 MISTs. For the sake of demonstration, we will use 100,000,000 MISTs, which is 0.1 SUI.

The output of the publish command is rather lengthy, so we will show and explain it in parts.

As you can see, when we run the publish command, the CLI first builds the package, then verifies the dependencies on-chain, and finally publishes the package. The output of the command is the transaction digest, which is a unique identifier of the transaction and can be used to query the transaction status.

The section titled TransactionData contains the information about the transaction we just sent. It features fields like sender , which is your address, the gas_budget set with the --gas-budget argument, and the Coin we used for payment. It also prints the Commands that were run by the CLI. In this example, the commands Publish and TransferObject were run - the latter transfers a special object UpgradeCap to the sender.

Transaction Effects contains the status of the transaction, the changes that the transaction made to the state of the network and the objects involved in the transaction.

If there were any events emitted, you would see them in this section. Our package does not use events, so the section is empty.

These are the changes to objects that transaction has made. In our case, we have created a new UpgradeCap object which is a special object that allows the sender to upgrade the package in the future, mutated the Gas object, and published a new package. Packages are also objects on Sui.

This last section contains changes to SUI Coins, in our case, we have spent around 0.015 SUI, which in MIST is 10,500,000. You can see it under the amount field in the output.

It is possible to specify the --json flag during publishing to get the output in JSON format. This is useful if you want to parse the output programmatically or store it for later use.

After the package is published on chain, we can interact with it. To do this, we need to find the address (object ID) of the package. It's under the Published Objects section of the Object Changes output. The address is unique for each package, so you will need to copy it from the output.

In this example, the address is:

Now that we have the address, we can interact with the package. In the next section, we will show how to interact with the package by sending transactions.

To demonstrate the interaction with the todo_list package, we will send a transaction to create a new list and add an item to it. Transactions are sent via the sui client ptb command, it allows using the Transaction Blocks at full capacity. The command may look big and complex, but we go through it step by step.

Before we construct the command, let's store the values we will use in the transaction. Replace the 0x4.... with the address of the package you have published. And MY_ADDRESS variable will be automatically set to your address from the CLI output.

Now to building an actual transaction. The transaction will consist of two parts: we will call the new function in the todo_list package to create a new list, and then we will transfer the list object to our account. The transaction will look like this:

In this command, we are using the ptb subcommand to build a transaction. Parameters that follow it define the actual commands and actions that the transaction will perform. The first two calls we make are utility calls to set the sender address to the command inputs and set the gas budget for the transaction.

Then we perform the actual call to a function in the package. We use the --move-call followed by the package ID, the module name, and the function name. In this case, we are calling the new function in the todo_list package.

The function that we defined actually returns a value, which we want need to store. We use the --assign command to give a name to the returned value. In this case, we are calling it list . And then we transfer the object to our account.

Once the command is constructed, you can run it in the terminal. If everything is correct, you should see the output similar to the one we had in previous sections. The output will contain the transaction digest, the transaction data, and the transaction effects.

The section that we want to focus on is the "Object Changes". More specifically, the "Created Objects" part of it. It contains the object ID, the type and the version of the TodoList that you have created. We will use this object ID to interact with the list.

In this example the object ID is 0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 . And the owner should be your account address. We achieved this by transferring the object to the sender in the last command of the transaction.

Another way to test that you have successfully created the list is to check the account objects.

It should have an object that looks similar to this:

The TodoList that we created in the previous step is an object that you can interact with as its owner. You can call functions defined in the todo_list module on this object. To demonstrate this, we will add an item to the list. First, we will add just one item, and in the second transaction we will add 3 and remove another one.

Double check that you have variables set up [from the previous step](), and then add one more variable for the list object.

Now we can construct the transaction to add an item to the list. The command will look like this:

In this command, we are calling the add function in the todo_list package. The function takes two arguments: the list object and the item to add. The item is a string, so we need to wrap it in single quotes. The command will add the item to the list.

If everything is correct, you should see the output similar to the one we had in previous sections. Now you can check the list object to see if the item was added.

The output should contain the item that you have added.

A JSON representation of the object can be obtained by adding the --json flag to the command.

You can chain multiple commands in a single transaction. This shows the power of Transaction Blocks! Using the same list object, we will add three more items and remove one. The command will look like this:

If previous commands were successful, this one should not be any different. You can check the list object to see if the items were added and removed. The JSON representation is a bit more readable!

Commands don't have to be in the same package or operate on the same object. Within a single transaction block, you can interact with multiple packages and objects. This is a powerful feature that allows you to build complex interactions on-chain!

In this guide, we have shown how to publish a package on the Move blockchain and interact with it using the Sui CLI. We have demonstrated how to create a new list object, add items to it, and remove them. We have also shown how to chain multiple commands in a single transaction block. This guide should give you a good starting point for building your own applications on the Sui blockchain!

## Create a New Package

Following the same flow as in [Hello, World!](), we will create a new package called todo_list .

```
bash $ sui move new todo_list
```

To speed things up and focus on the application logic, we will provide the code for the todo list application. Replace the contents of the sources/todo_list.move file with the following code:

Note: while the contents may seem overwhelming at first, we will break it down in the following sections. Try to focus on what's at hand right now.

```bash /// Module: todo_list module todo_list::todo_list;

use std::string::String;

/// List of todos. Can be managed by the owner and shared with others. public struct TodoList has key, store { id: UID, items: vector }

/// Create a new todo list. public fun new(ctx: &mut TxContext): TodoList { let list = TodoList { id: object::new(ctx), items: vector[] };

```
(list)
```

}

/// Add a new todo item to the list. public fun add(list: &mut TodoList, item: String) { list.items.push_back(item); }

/// Remove a todo item from the list by index. public fun remove(list: &mut TodoList, index: u64): String { list.items.remove(index) }

/// Delete the list and the capability to manage it. public fun delete(list: TodoList) { let TodoList { id, items: _ } = list; id.delete(); }

/// Get the number of items in the list. public fun length(list: &TodoList): u64 { list.items.length() } ```

To make sure that we did everything correctly, let's build the package by running the sui move build command. If everything is correct, you should see the output similar to the following:

```
bash $ sui move build UPDATING GIT DEPENDENCY https://github.com/MystenLabs/sui.git INCLUDING DEPENDENCY Bridge INCLUDING DEPENDENCY DeepBook INCLUDING
DEPENDENCY SuiSystem INCLUDING DEPENDENCY Sui INCLUDING DEPENDENCY MoveStdlib BUILDING todo_list
~~~

If there are no errors following this output, you have successfully built the package. If there are errors, make sure that:

There are not many other reasons for the code to fail at this stage. But if you are still having issues, try looking up the structure of the package in [this location]() .

If you already have an account set up, you can skip this step.

To publish and interact with the package, we need to set up an account. While developing, the best option for doing so is to run your own [Local Network]() . For now you just need to run RUST_LOG="off,sui_node=info" sui start --with-faucet --force-regenesis . The Sui Local Network will run on port 9000 of your machine, so make sure that the port isn't being used by any other application.

If you are doing it for the first time, you will need to create a new account. To do this, run the sui client command, then the CLI will prompt you with multiple questions. The answers are marked below with > :

```bash $ sui client Config file ["/path/to/home/.sui/sui_config/client.yaml"] doesn't exist, do you want to connect to a Sui Full node server [y/N]?

> y Sui Full node server URL (Defaults to Sui Testnet if not specified) : http://127.0.0.1:9000 Environment alias for [http://127.0.0.1:9000] : localnet Select key scheme to generate keypair (0 for ed25519, 1 for secp256k1, 2: for secp256r1): 0 ```

After you have answered the questions, the CLI will generate a new keypair and save it to the configuration file. You can now use this account to interact with the network.

To check that we have the account set up correctly, run the sui client active-address command:

~~~
bash $ sui client active-address 0x....
```

The command will output the address of your account, it starts with 0x followed by 64 characters.

In devnet and testnet environments, the CLI provides a way to request coins to your account, so you can interact with the network. To request coins, run the sui client faucet command:

```
bash $ sui client faucet Request successful. It can take up to 1 minute to get the coin. Run sui client gas to check your gas coins.
```

After waiting a little bit, you can check that the Coin object was sent to your account by running the sui client balance command:

```
bash $ sui client balance ╭────────────────────────────────────────╮ │ Balance of coins owned by this address │
├────────────────────────────────────────┤ │ ╭──────────────────────────────────╮ │ │ │ coin balance (raw) balance │ │ │
├──────────────────────────────────┤ │ │ │ Sui 1000000000 1.00 SUI │ │ │ ╰──────────────────────────────────╯ │
```
```

```
╰────────────────────────────────────────╯
```

Alternatively, you can query objects owned by your account, by running the sui client objects command. The actual output will be different, because the object ID is unique, and so is digest, but the structure will be similar:

```
bash $ sui client objects ╭────────────────────────────────────────────────────────────────────────────────╮ │
╭──────────────┬──────────────────────────────────────────────────────────────────────╮ │ │ │ objectId │
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ │ version │ 4 │ │ │ │ digest │ nA68oa8gab/CdIRw+240wze8u0P+sRe4vcisbENcR4U= │ │
│ │ objectType │ 0x0000..0002::coin::Coin │ │ │ ╰──────────────┴──────────────────────────────────────────────────────────────────────╯ │
╰────────────────────────────────────────────────────────────────────────────────╯
```

Now that we have the account set up and the coins in the account, we can interact with the network. We will start by publishing the package to the network.

To publish the package to the network, we will use the sui client publish command. The command will automatically build the package and use its bytecode to publish in a single transaction.

We are using the --gas-budget argument during publishing. It specifies how much gas we are willing to spend on the transaction. We won't touch on this topic in this section, but it's important to know that every transaction in Sui costs gas, and the gas is paid in SUI coins.

The gas-budget is specified in MISTs . 1 SUI equals 10^9 MISTs. For the sake of demonstration, we will use 100,000,000 MISTs, which is 0.1 SUI.

```bash

## run this from the `todo_list` folder

$ sui client publish --gas-budget 100000000

## alternatively, you can specify path to the package

$ sui client publish --gas-budget 100000000 todo_list ```

The output of the publish command is rather lengthy, so we will show and explain it in parts.

```
bash $ sui client publish --gas-budget 100000000 UPDATING GIT DEPENDENCY https://github.com/MystenLabs/sui.git INCLUDING DEPENDENCY Bridge INCLUDING
DEPENDENCY DeepBook INCLUDING DEPENDENCY SuiSystem INCLUDING DEPENDENCY Sui INCLUDING DEPENDENCY MoveStdlib BUILDING todo_list Successfully verified
dependencies on-chain against source. Transaction Digest: GpcDV6JjjGQMRwHpEz582qsd5MpCYgSwrDAq1JXcpFjW
```

As you can see, when we run the publish command, the CLI first builds the package, then verifies the dependencies on-chain, and finally publishes the package. The output of the command is the transaction digest, which is a unique identifier of the transaction and can be used to query the transaction status.

The section titled TransactionData contains the information about the transaction we just sent. It features fields like sender , which is your address, the gas_budget set with the --gas-budget argument, and the Coin we used for payment. It also prints the Commands that were run by the CLI. In this example, the commands Publish and TransferObject were run - the latter transfers a special object UpgradeCap to the sender.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────────────────────╮ │ Transaction Data │
├──────────────────────────────────────────────────────────────────────────────────────────────────────────────────┤ │ Sender:
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Owner: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │
Gas Budget: 100000000 MIST │ │ Gas Price: 1000 MIST │ │ Gas Payment: │ │ ┌── │ │ │ ID: 0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de
│ │ │ Version: 7 │ │ │ Digest: AXYPnups8A5J6pkvLa6RekX2ye3qur66EZ88mEbaUDQ1 │ │ └── │ │ │ │ Transaction Kind: Programmable │ │
╭────────────────────────────────────────────────────────────────────────────────────────────────╮ │ │ │ Input Objects │ │ │
├────────────────────────────────────────────────────────────────────────────────────────────────┤ │ │ │ 0 Pure Arg: Type: address, Value:
"0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" │ │ │
╰────────────────────────────────────────────────────────────────────────────────────────────────╯ │ │
╭────────────────────────────────────────────────────────────────────╮ │ │ │ Commands │ │ │
├────────────────────────────────────────────────────────────────────┤ │ │ │ 0 Publish: │ │ │ │ ┌ │ │ │ │ │ Dependencies: │ │ │ │ │
0x0000000000000000000000000000000000000000000000000000000000000001 │ │ │ │ │ 0x0000000000000000000000000000000000000000000000000000000000000002 │ │ │ │ └
│ │ │ │ │ │ │ │ 1 TransferObjects: │ │ │ │ ┌ │ │ │ │ │ Arguments: │ │ │ │ │ Result 0 │ │ │ │ │ Address: Input 0 │ │ │ │ └ │ │ │
╰────────────────────────────────────────────────────────────────────╯ │ │ │ │ Signatures: │ │
gebjSbVwZwTkizfYg2XIuzdx+d66VxFz8EmVaisVFiV3GkDay6L+hQG3n2CQ1hrWphP6ZLc7bd1WRq4ss+hQAQ== │ │ │
╰──────────────────────────────────────────────────────────────────────────────────────────────────────╯
```

Transaction Effects contains the status of the transaction, the changes that the transaction made to the state of the network and the objects involved in the transaction.

```
bash ╭────────────────────────────────────────────────────────────────────────────────────────────────╮ │ Transaction Effects │
├──────────────────────────────────────────────────────────────────────────────────────────────────────┤ │ Digest:
GpcDV6JjjGQMRwHpEz582qsd5MpCYgSwrDAq1JXcpFjW │ │ Status: Success │ │ Executed Epoch: 411 │ │ │ │ Created Objects: │ │ ┌── │ │ │ ID:
0x160f7856e13b27e5a025112f361370f4efc2c2659cb0023f1e99a8a84d1652f3 │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 8y6bhwvQrGJHDckUZmj2HDAjfkyVqHohhvY1Fvzyj7ec │ │ └──
│ │ ┌── │ │ │ ID: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ Owner: Immutable │ │ │ Version: 1 │ │ │ Digest:
Ein91NF2hc3qC4XYoMUFMfin9U23xQmDAdEMSHLae7MK │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ID:
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └──
│ │ Gas Object: │ │ ┌── │ │ │ ID: 0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └──
│ │ Gas Cost Summary: │ │ Storage Cost: 10404400 MIST │ │ Computation Cost: 1000000 MIST │ │ Storage Rebate: 978120 MIST │ │ Non-refundable Storage Fee:
9880 MIST │ │ │ │ Transaction Dependencies: │ │ 7Ukrc5GqdFqTA41wvWgreCdHn2vRLfgQ3YMFkdks72Vk │ │ 7d4amuHGhjtYKujEs9YkJARzNEn4mRbWWv3fn4cdKdyh │
╰──────────────────────────────────────────────────────────────────────────────────────────────────────╯
```

If there were any events emitted, you would see them in this section. Our package does not use events, so the section is empty.

```
bash ╭─────────────────────────────╮ │ No transaction block events │ ╰─────────────────────────────╯
```

These are the changes to objects that transaction has made. In our case, we have created a new UpgradeCap object which is a special object that allows the sender to upgrade the package in the future, mutated the Gas object, and published a new package. Packages are also objects on Sui.

```
bash ╭────────────────────────────────────────────────────────────────────────────────────────────────╮ │ Object Changes │
├──────────────────────────────────────────────────────────────────────────────────────────────────────┤ │ Created Objects: │ │ ┌── │ │ │ ObjectID:
0x160f7856e13b27e5a025112f361370f4efc2c2659cb0023f1e99a8a84d1652f3 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::package::UpgradeCap │ │ │ Version: 8
│ │ │ Digest: 8y6bhwvQrGJHDckUZmj2HDAjfkyVqHohhvY1Fvzyj7ec │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │
Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └── │ │ Published Objects: │ │ ┌── │ │ │ PackageID:
0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ Version: 1 │ │ │ Digest: Ein91NF2hc3qC4XYoMUFMfin9U23xQmDAdEMSHLae7MK │ │ │
Modules: todo_list │ │ └── │ ╰────────────────────────────────────────────────────────────────────────────────────╯
```

This last section contains changes to SUI Coins, in our case, we have spent around 0.015 SUI, which in MIST is 10,500,000. You can see it under the amount field in the output.

```
bash ╭────────────────────────────────────────────────────────────────────────────────────────────────╮ │ Balance Changes │
├──────────────────────────────────────────────────────────────────────────────────────────────────────┤ │ ┌── │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ CoinType: 0x2::sui::SUI │ │ │ Amount: -10426280 │ │ └── │
╰──────────────────────────────────────────────────────────────────────────────────────────────────────╯
```

It is possible to specify the --json flag during publishing to get the output in JSON format. This is useful if you want to parse the output programmatically or store it for later use.

```
bash $ sui client publish --gas-budget 100000000 --json
```

After the package is published on chain, we can interact with it. To do this, we need to find the address (object ID) of the package. It's under the Published Objects section of the Object Changes output.
```

The address is unique for each package, so you will need to copy it from the output.

In this example, the address is:

```
bash 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe
```

Now that we have the address, we can interact with the package. In the next section, we will show how to interact with the package by sending transactions.

To demonstrate the interaction with the todo_list package, we will send a transaction to create a new list and add an item to it. Transactions are sent via the sui client ptb command, it allows using the [Transaction Blocks](#) at full capacity. The command may look big and complex, but we go through it step by step.

Before we construct the command, let's store the values we will use in the transaction. Replace the 0x4.... with the address of the package you have published. And MY_ADDRESS variable will be automatically set to your address from the CLI output.

```
bash $ export PACKAGE_ID=0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe $ export MY_ADDRESS=$(sui client active-address)
```

Now to building an actual transaction. The transaction will consist of two parts: we will call the new function in the todo_list package to create a new list, and then we will transfer the list object to our account. The transaction will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --assign sender @$MY_ADDRESS \ --move-call $PACKAGE_ID::todo_list::new \ --assign list \ --transfer-objects "[list]" sender
```

In this command, we are using the ptb subcommand to build a transaction. Parameters that follow it define the actual commands and actions that the transaction will perform. The first two calls we make are utility calls to set the sender address to the command inputs and set the gas budget for the transaction.

~~~bash

## sets the gas budget for the transaction

--gas-budget 100000000 \n

## registers a variable "sender=@..."

--assign sender @$MY_ADDRESS \n ```

Then we perform the actual call to a function in the package. We use the --move-call followed by the package ID, the module name, and the function name. In this case, we are calling the new function in the todo_list package.

```bash

## calls the "new" function in the "todo_list" package under the $PACKAGE_ID address

--move-call $PACKAGE_ID::todo_list::new ```

The function that we defined actually returns a value, which we want need to store. We use the --assign command to give a name to the returned value. In this case, we are calling it list . And then we transfer the object to our account.

```bash --move-call $PACKAGE_ID::todo_list::new \

## assigns the result of the "new" function to the "list" variable (from the previous step)

--assign list \

## transfers the object to the sender

--transfer-objects "[list]" sender ```

Once the command is constructed, you can run it in the terminal. If everything is correct, you should see the output similar to the one we had in previous sections. The output will contain the transaction digest, the transaction data, and the transaction effects.

```bash Transaction Digest: BJwYEnuuMzU4Y8cTwMoJbbQA6cLwPmwxvsRpSmvThoK8

╭──────────────────────────────────────────────────────────────────────────────╮

│ Transaction Data │

├──────────────────────────────────────────────────────────────────────────────┤

│ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Owner: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Budget: 100000000 MIST │ │ Gas Price: 1000 MIST │ │ Gas Payment: │ │ ┌── │ │ │ ID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Version: 22 │ │ │ Digest: DiBrBMshDiD9cThpaEgpcYSF76uV4hCoE1qRyQ3rnYCB │ │ └── │ │ │ │ Transaction Kind: Programmable │ │

╭──────────────────────────────────────────────────────────────────────────╮ │ │

│ Input Objects │ │ │

├──────────────────────────────────────────────────────────────────────────┤ │ │

│ 0 Pure Arg: Type: address, Value: "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" │ │ │

╰──────────────────────────────────────────────────────────────────────────╯ │ │

╭──────────────────────────────────────────────────────────────╮ │ │ │ Commands │ │ │

├──────────────────────────────────────────────────────────────┤ │ │ │ 0 MoveCall: │ │ │ │ ┌ │ │ │ │ │

Function: new │ │ │ │ │ Module: todo_list │ │ │ │ │ Package: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ │ └ │ │ │ │ │ │ │ │ 1 TransferObjects: │ │ │ │ ┌ │ │ │ │ │ Arguments: │ │ │ │ │ Result 0 │ │ │ │ │ Address: Input 0 │ │ │ │ └ │ │ │

╰──────────────────────────────────────────────────────────────╯ │ │ │ │ Signatures: │ │

C5Lie4dtP5d3OkKzFBa+xM0BiNoB/A4ItthDCRTRBUrEE+jXeNs7mP4AuGwi3nzfTskh29+R1j1Kba4Wdy3QDA== │ │ │

╰──────────────────────────────────────────────────────────────────────────────╯

╭──────────────────────────────────────────────────────────────────────────╮ │ Transaction Effects │ ├──────────────────────────────────────────────────────────────────────────────┤ │

Digest: BJwYEnuuMzU4Y8cTwMoJbbQA6cLwPmwxvsRpSmvThoK8 │ │ Status: Success │ │ Executed Epoch: 1213 │ │ │ │ Created Objects: │ │ ┌── │ │ │ ID: 0x74973c4ea2e78dc409f60481e23761cee68a48156df93a93fbcceb77d1cacdf6 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 23 │ │ │ Digest: DuHTozDHMsuA7cFnWRQ1Gb8FQghAEBaj3inasJxqYq1c │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │ │ Gas Object: │ │ ┌── │ │ │ ID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │ │ Gas Cost Summary: │ │ Storage Cost: 2318000 MIST │ │ Computation Cost: 1000000 MIST │ │ Storage Rebate: 978120 MIST │ │ Non-refundable Storage Fee: 9880 MIST │ │ │ │ Transaction Dependencies: │ │ FSz2fYXmKqTf77mFXNq5JK7cKY8agWja7V5yDKEgL8c3 │ │ GgMZKTt482DYApbAZkPDtdssGHZLbxgjm2uMXhzJax8Q │

╰──────────────────────────────────────────────────────────────────────────╯

╭─────────────────────────────╮ │ No transaction block events │ ╰─────────────────────────────╯
~~~

```
╭──────────────────────────────────────────────────────────────────────────────╮ │ Object
Changes │
├──────────────────────────────────────────────────────────────────────────────┤ │ Created
Objects: │ │ ┌── │ │ │ ObjectID: 0x74973c4ea2e78dc409f60481e23761cee68a48156df93a93fbcceb77d1cacdf6 │ │ │ Sender:
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │
│ │ ObjectType: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ Version: 23 │ │ │ Digest:
DuHTozDHMsuA7cFnWRQ1Gb8FQghAEBaj3inasJxqYq1c │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ Owner:
Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │ Version: 23 │ │ │ Digest:
82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │
╰──────────────────────────────────────────────────────────────────────────────╯

╭────────────────────────────────────────────────────────────────────────────╮ │ Balance
Changes │ │├──────────────────────────────────────────────────────────────────────────┤ │
┌── │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ CoinType: 0x2::sui::SUI │ │ │ Amount: -2339880 │ │ └── │
╰────────────────────────────────────────────────────────────────────────────╯ ```
```

The section that we want to focus on is the "Object Changes". More specifically, the "Created Objects" part of it. It contains the object ID, the type and the version of the TodoList that you have created. We will use this object ID to interact with the list.

```
bash ╭────────────────────────────────────────────────────────────────────────╮ │ Object Changes │
├────────────────────────────────────────────────────────────────────────┤ │ Created Objects: │ │ ┌── │ │ │ ObjectID:
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType:
0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ Version: 22 │ │ │ Digest:
HyWdUpjuhjLY38dLpg6KPHQ3bt4BqQAbdF5gB8HQdEqG │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │
Version: 22 │ │ │ Digest: DiBrBMshDiD9cThpaEgpcYSF76uV4hCoE1qRyQ3rnYCB │ │ └── │
╰────────────────────────────────────────────────────────────────────────╯
```

In this example the object ID is 0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 . And the owner should be your account address. We achieved this by transferring the object to the sender in the last command of the transaction.

Another way to test that you have successfully created the list is to check the account objects.

```
bash $ sui client objects
```

It should have an object that looks similar to this:

```
bash ╭ ... ╮ │ ╭───────────┬──────────────────────────────────────────────────────╮ │ │ │ objectId │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ │ version │ 22 │ │ │ │ digest │ /DUEiCLkaNSgzpZSq2vSV0auQQEQhyH9occq9grMBZM= │ │
│ │ objectType │ 0x468d..29fe::todo_list::TodoList │ │ │ ╰───────────┴──────────────────────────────────────────────────────╯ │ │ ... │
```

The TodoList that we created in the previous step is an object that you can interact with as its owner. You can call functions defined in the todo_list module on this object. To demonstrate this, we will add an item to the list. First, we will add just one item, and in the second transaction we will add 3 and remove another one.

Double check that you have variables set up [from the previous step](), and then add one more variable for the list object.

```
bash $ export LIST_ID=0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553
```

Now we can construct the transaction to add an item to the list. The command will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Finish the Hello, Sui chapter'"
```

In this command, we are calling the add function in the todo_list package. The function takes two arguments: the list object and the item to add. The item is a string, so we need to wrap it in single quotes. The command will add the item to the list.

If everything is correct, you should see the output similar to the one we had in previous sections. Now you can check the list object to see if the item was added.

```
bash $ sui client object $LIST_ID
```

The output should contain the item that you have added.

```
bash ╭─────────────┬──────────────────────────────────────────────────────────────────────╮ │ objectId │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ version │ 24 │ │ digest │ FGcXH8MGpMs5BdTnC62CQ3VLAwwexYg2id5DKU7Jr9aQ │ │ objType
│ 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ owner │
╭───────────────┬──────────────────────────────────────────────────────╮ │ │ │ │ AddressOwner │
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ │
╰───────────────┴──────────────────────────────────────────────────────╯ │ │ prevTx │ EJVK6FEHtfTdCuGkNsU1HcrmUBEN6H6jshfcptnw8Yt1 │ │
storageRebate │ 1558000 │ │ content │ ╭───────────────────┬──────────────────────────────────────────────────────╮
│ │ │ │ dataType │ moveObject │ │ │ │ │ type │ 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ │ │
hasPublicTransfer │ true │ │ │ │ │ fields │ ╭────────┬──────────────────────────────────────────────╮ │ │ │ │ │ │ │ id │
╭──────┬──────────────────────────────────────────────────────────────╮ │ │ │ │ │ │ │ │ │ │ id │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ │ │ │ │ │ │ │
╰──────┴──────────────────────────────────────────────────────────────╯ │ │ │ │ │ │ │ │ items │ ╭──────────────────────────╮ │ │ │ │ │ │ │
│ │ │ finish the Hello, Sui chapter │ │ │ │ │ │ │ │ │ │ ╰──────────────────────────╯ │ │ │ │ │ │ │
╰─────────┴────────────────────────────────────────────────────────╯ │ │ │ │
╰───────────────────┴──────────────────────────────────────────────────────╯ │
╰─────────────┴──────────────────────────────────────────────────────────────────────╯
```

A JSON representation of the object can be obtained by adding the --json flag to the command.

```
bash $ sui client object $LIST_ID --json
```

```
bash { "objectId": "0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553", "version": "24", "digest":
"FGcXH8MGpMs5BdTnC62CQ3VLAwwexYg2id5DKU7Jr9aQ", "type": "0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList",
"owner": { "AddressOwner": "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" }, "previousTransaction":
"EJVK6FEHtfTdCuGkNsU1HcrmUBEN6H6jshfcptnw8Yt1", "storageRebate": "1558000", "content": { "dataType": "moveObject", "type":
"0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList", "hasPublicTransfer": true, "fields": { "id": { "id":
"0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553" }, "items": ["Finish the Hello, Sui chapter"] } } }
```

You can chain multiple commands in a single transaction. This shows the power of Transaction Blocks! Using the same list object, we will add three more items and remove one. The command will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Finish Concepts chapter'" \ --move-call
$PACKAGE_ID::todo_list::add @$LIST_ID "'Read the Move Basics chapter'" \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Learn about Object Model'" \
--move-call $PACKAGE_ID::todo_list::remove @$LIST_ID 0
```

If previous commands were successful, this one should not be any different. You can check the list object to see if the items were added and removed. The JSON representation is a bit more readable!

```
bash sui client object $LIST_ID --json
```

```
bash { "objectId": "0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553", "version": "25", "digest":
"EDTXDsteqPGAGu4zFAj5bbQGTkucWk4hhuUquk39enGA", "type": "0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList",
"owner": { "AddressOwner": "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" }, "previousTransaction":
"7SXLGBSh31jv8G7okQ9mEgnw5MnTfvzzHEHpWf3Sa9gY", "storageRebate": "1922800", "content": { "dataType": "moveObject", "type":
"0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList", "hasPublicTransfer": true, "fields": { "id": { "id":
"0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553" }, "items": [ "Finish Concepts chapter", "Read the Move Basics chapter", "Learn
```

```
about Object Model" ] } } }
```

Commands don't have to be in the same package or operate on the same object. Within a single transaction block, you can interact with multiple packages and objects. This is a powerful feature that allows you to build complex interactions on-chain!

In this guide, we have shown how to publish a package on the Move blockchain and interact with it using the Sui CLI. We have demonstrated how to create a new list object, add items to it, and remove them. We have also shown how to chain multiple commands in a single transaction block. This guide should give you a good starting point for building your own applications on the Sui blockchain!

## Add the code

To speed things up and focus on the application logic, we will provide the code for the todo list application. Replace the contents of the sources/todo_list.move file with the following code:

Note: while the contents may seem overwhelming at first, we will break it down in the following sections. Try to focus on what's at hand right now.

```bash /// Module: todo_list module todo_list::todo_list;

use std::string::String;

/// List of todos. Can be managed by the owner and shared with others. public struct TodoList has key, store { id: UID, items: vector }

/// Create a new todo list. public fun new(ctx: &mut TxContext): TodoList { let list = TodoList { id: object::new(ctx), items: vector[] };

```
(list)
```

}

/// Add a new todo item to the list. public fun add(list: &mut TodoList, item: String) { list.items.push_back(item); }

/// Remove a todo item from the list by index. public fun remove(list: &mut TodoList, index: u64): String { list.items.remove(index) }

/// Delete the list and the capability to manage it. public fun delete(list: TodoList) { let TodoList { id, items: _ } = list; id.delete(); }

/// Get the number of items in the list. public fun length(list: &TodoList): u64 { list.items.length() } ```

To make sure that we did everything correctly, let's build the package by running the sui move build command. If everything is correct, you should see the output similar to the following:

```
bash $ sui move build UPDATING GIT DEPENDENCY https://github.com/MystenLabs/sui.git INCLUDING DEPENDENCY Bridge INCLUDING DEPENDENCY DeepBook INCLUDING
DEPENDENCY SuiSystem INCLUDING DEPENDENCY Sui INCLUDING DEPENDENCY MoveStdlib BUILDING todo_list
~~~

If there are no errors following this output, you have successfully built the package. If there are errors, make sure that:

There are not many other reasons for the code to fail at this stage. But if you are still having issues, try looking up the structure of the package in this location .

If you already have an account set up, you can skip this step.

To publish and interact with the package, we need to set up an account. While developing, the best option for doing so is to run your own Local Network . For now you just need to run RUST_LOG="off,sui_node=info" sui start --with-faucet --force-regenesis . The Sui Local Network will run on port 9000 of your machine, so make sure that the port isn't being used by any other application.

If you are doing it for the first time, you will need to create a new account. To do this, run the sui client command, then the CLI will prompt you with multiple questions. The answers are marked below with > :

```bash $ sui client Config file ["/path/to/home/.sui/sui_config/client.yaml"] doesn't exist, do you want to connect to a Sui Full node server [y/N]?

> y Sui Full node server URL (Defaults to Sui Testnet if not specified) : http://127.0.0.1:9000 Environment alias for [http://127.0.0.1:9000] : localnet Select key scheme to generate keypair (0 for ed25519, 1 for secp256k1, 2: for secp256r1): 0 ```

After you have answered the questions, the CLI will generate a new keypair and save it to the configuration file. You can now use this account to interact with the network.

To check that we have the account set up correctly, run the sui client active-address command:

~~~
bash $ sui client active-address 0x....
```

The command will output the address of your account, it starts with 0x followed by 64 characters.

In devnet and testnet environments, the CLI provides a way to request coins to your account, so you can interact with the network. To request coins, run the sui client faucet command:

```
bash $ sui client faucet Request successful. It can take up to 1 minute to get the coin. Run sui client gas to check your gas coins.
```

After waiting a little bit, you can check that the Coin object was sent to your account by running the sui client balance command:

```
bash $ sui client balance ╭────────────────────────────────────────╮ │ Balance of coins owned by this address │
├────────────────────────────────────────┤ │ ╭──────────────────────────────────────╮ │ │ │ coin balance (raw) balance │ │ │
├──────────────────────────────────────┤ │ │ │ Sui 1000000000 1.00 SUI │ │ │ ╰──────────────────────────────────────╯ │
╰────────────────────────────────────────╯
```

Alternatively, you can query objects owned by your account, by running the sui client objects command. The actual output will be different, because the object ID is unique, and so is digest, but the structure will be similar:

```
bash $ sui client objects ╭───────────────────────────────────────────────────────────────────────────────────────╮ │
╭────────────┬──────────────────────────────────────────────────────────────────────╮ │ │ │ objectId │
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ │ version │ 4 │ │ │ │ digest │ nA68oa8gab/CdIRw+240wze8u0P+sRe4vcisbENcR4U= │ │
│ │ objectType │ 0x0000..0002::coin::Coin │ │ │ ╰────────────┴──────────────────────────────────────────────────────────────────────╯ │
╰───────────────────────────────────────────────────────────────────────────────────────╯
~~~

Now that we have the account set up and the coins in the account, we can interact with the network. We will start by publishing the package to the network.

To publish the package to the network, we will use the sui client publish command. The command will automatically build the package and use its bytecode to publish in a single transaction.

We are using the --gas-budget argument during publishing. It specifies how much gas we are willing to spend on the transaction. We won't touch on this topic in this section, but it's important to know that every transaction in Sui costs gas, and the gas is paid in SUI coins.

The gas-budget is specified in MISTs . 1 SUI equals 10^9 MISTs. For the sake of demonstration, we will use 100,000,000 MISTs, which is 0.1 SUI.

```bash

# run this from the `todo_list` folder

$ sui client publish --gas-budget 100000000

# alternatively, you can specify path to the package
~~~

$ sui client publish --gas-budget 100000000 todo_list ```

The output of the publish command is rather lengthy, so we will show and explain it in parts.

```
bash $ sui client publish --gas-budget 100000000 UPDATING GIT DEPENDENCY https://github.com/MystenLabs/sui.git INCLUDING DEPENDENCY Bridge INCLUDING
DEPENDENCY DeepBook INCLUDING DEPENDENCY SuiSystem INCLUDING DEPENDENCY Sui INCLUDING DEPENDENCY MoveStdlib BUILDING todo_list Successfully verified
dependencies on-chain against source. Transaction Digest: GpcDV6JjjGQMRwHpEz582qsd5MpCYgSwrDAq1JXcpFjW
```

As you can see, when we run the publish command, the CLI first builds the package, then verifies the dependencies on-chain, and finally publishes the package. The output of the command is the transaction digest, which is a unique identifier of the transaction and can be used to query the transaction status.

The section titled TransactionData contains the information about the transaction we just sent. It features fields like sender , which is your address, the gas_budget set with the --gas-budget argument, and the Coin we used for payment. It also prints the Commands that were run by the CLI. In this example, the commands Publish and TransferObject were run - the latter transfers a special object UpgradeCap to the sender.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────────────────╮ │ Transaction Data │
├──────────────────────────────────────────────────────────────────────────────────────────────────┤ │ Sender:
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Owner: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │
Gas Budget: 100000000 MIST │ │ Gas Price: 1000 MIST │ │ Gas Payment: │ │ ┌── │ │ │ ID: 0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de
│ │ │ Version: 7 │ │ │ Digest: AXYPnups8A5J6pkvLa6RekX2ye3qur66EZ88mEbaUDQ1 │ │ └── │ │ │ │ Transaction Kind: Programmable │ │
╭──────────────────────────────────────────────────────────────────────────────────────────────╮ │ │ │ Input Objects │ │ │
├──────────────────────────────────────────────────────────────────────────────────────────────┤ │ │ │ 0 Pure Arg: Type: address, Value:
"0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" │ │ │
╰──────────────────────────────────────────────────────────────────────────────────────────────╯ │ │
╭──────────────────────────────────────────────────────────────────────────╮ │ │ │ Commands │ │ │
├──────────────────────────────────────────────────────────────────────────┤ │ │ │ 0 Publish: │ │ │ │ ┌ │ │ │ │ │ Dependencies: │ │ │ │ │
0x0000000000000000000000000000000000000000000000000000000000000001 │ │ │ │ │ 0x0000000000000000000000000000000000000000000000000000000000000002 │ │ │ │ └
│ │ │ │ │ │ │ │ 1 TransferObjects: │ │ │ │ ┌ │ │ │ │ │ Arguments: │ │ │ │ │ Result 0 │ │ │ │ │ Address: Input 0 │ │ │ │ └ │ │ │
╰──────────────────────────────────────────────────────────────────────────╯ │ │ │ │ Signatures: │ │
gebjSbVwZwTkizfYg2XIuzdx+d66VxFz8EmVaisVFiV3GkDay6L+hQG3n2CQ1hrWphP6ZLc7bd1WRq4ss+hQAQ== │ │ │
╰──────────────────────────────────────────────────────────────────────────────────────────────────╯
```

Transaction Effects contains the status of the transaction, the changes that the transaction made to the state of the network and the objects involved in the transaction.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────────╮ │ Transaction Effects │
├──────────────────────────────────────────────────────────────────────────────────────────┤ │ Digest:
GpcDV6JjjGQMRwHpEz582qsd5MpCYgSwrDAq1JXcpFjW │ │ Status: Success │ │ Executed Epoch: 411 │ │ │ │ Created Objects: │ │ ┌── │ │ │ ID:
0x160f7856e13b27e5a025112f361370f4efc2c2659cb0023f1e99a8a84d1652f3 │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 8y6bhwvQrGJHDckUZmj2HDAjfkyVqHohhvY1Fvzyj7ec │ │ └──
│ │ ┌── │ │ │ ID: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ Owner: Immutable │ │ │ Version: 1 │ │ │ Digest:
Ein91NF2hc3qC4XYoMUFMfin9U23xQmDAdEMSHLae7MK │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ID:
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └──
│ │ Gas Object: │ │ ┌── │ │ │ ID: 0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └──
│ │ Gas Cost Summary: │ │ Storage Cost: 10404400 MIST │ │ Computation Cost: 1000000 MIST │ │ Storage Rebate: 978120 MIST │ │ Non-refundable Storage Fee:
9880 MIST │ │ │ │ Transaction Dependencies: │ │ 7Ukrc5GqdFqTA41wvWgreCdHn2vRLfgQ3YMFkdks72Vk │ │ 7d4amuHGhjtYKujEs9YkJARzNEn4mRbWWv3fn4cdKdyh │
╰──────────────────────────────────────────────────────────────────────────────────────────╯
```

If there were any events emitted, you would see them in this section. Our package does not use events, so the section is empty.

```
bash ╭─────────────────────────────╮ │ No transaction block events │ ╰─────────────────────────────╯
```

These are the changes to objects that transaction has made. In our case, we have created a new UpgradeCap object which is a special object that allows the sender to upgrade the package in the future, mutated the Gas object, and published a new package. Packages are also objects on Sui.

```
bash ╭─────────────────────────────────────────────────────────────────────────────────────────╮ │ Object Changes │
├─────────────────────────────────────────────────────────────────────────────────────────┤ │ Created Objects: │ │ ┌── │ │ │ ObjectID:
0x160f7856e13b27e5a025112f361370f4efc2c2659cb0023f1e99a8a84d1652f3 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::package::UpgradeCap │ │ │ Version: 8
│ │ │ Digest: 8y6bhwvQrGJHDckUZmj2HDAjfkyVqHohhvY1Fvzyj7ec │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │
Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └── │ │ Published Objects: │ │ ┌── │ │ │ PackageID:
0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ Version: 1 │ │ │ Digest: Ein91NF2hc3qC4XYoMUFMfin9U23xQmDAdEMSHLae7MK │ │ │
Modules: todo_list │ │ └── │ ╰─────────────────────────────────────────────────────────────────────────────────────────╯
```

This last section contains changes to SUI Coins, in our case, we have spent around 0.015 SUI, which in MIST is 10,500,000. You can see it under the amount field in the output.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────────╮ │ Balance Changes │
├──────────────────────────────────────────────────────────────────────────────────────────┤ │ ┌── │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ CoinType: 0x2::sui::SUI │ │ │ Amount: -10426280 │ │ └── │
╰──────────────────────────────────────────────────────────────────────────────────────────╯
```

It is possible to specify the --json flag during publishing to get the output in JSON format. This is useful if you want to parse the output programmatically or store it for later use.

```
bash $ sui client publish --gas-budget 100000000 --json
```

After the package is published on chain, we can interact with it. To do this, we need to find the address (object ID) of the package. It's under the Published Objects section of the Object Changes output. The address is unique for each package, so you will need to copy it from the output.

In this example, the address is:

```
bash 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe
```

Now that we have the address, we can interact with the package. In the next section, we will show how to interact with the package by sending transactions.

To demonstrate the interaction with the todo_list package, we will send a transaction to create a new list and add an item to it. Transactions are sent via the sui client ptb command, it allows using the Transaction Blocks at full capacity. The command may look big and complex, but we go through it step by step.

Before we construct the command, let's store the values we will use in the transaction. Replace the 0x4.... with the address of the package you have published. And MY_ADDRESS variable will be automatically set to your address from the CLI output.

```
bash $ export PACKAGE_ID=0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe $ export MY_ADDRESS=$(sui client active-address)
```

Now to building an actual transaction. The transaction will consist of two parts: we will call the new function in the todo_list package to create a new list, and then we will transfer the list object to our account. The transaction will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --assign sender @$MY_ADDRESS \ --move-call $PACKAGE_ID::todo_list::new \ --assign list \ --transfer-
objects "[list]" sender
```

In this command, we are using the ptb subcommand to build a transaction. Parameters that follow it define the actual commands and actions that the transaction will perform. The first two calls we make are utility calls to set the sender address to the command inputs and set the gas budget for the transaction.

```bash

# sets the gas budget for the transaction
```

--gas-budget 100000000 \n

## registers a variable "sender=@..."

--assign sender @$MY_ADDRESS \n ```

Then we perform the actual call to a function in the package. We use the --move-call followed by the package ID, the module name, and the function name. In this case, we are calling the new function in the todo_list package.

~~~bash

## calls the "new" function in the "todo_list" package under the $PACKAGE_ID address

--move-call $PACKAGE_ID::todo_list::new ```

The function that we defined actually returns a value, which we want need to store. We use the --assign command to give a name to the returned value. In this case, we are calling it list . And then we transfer the object to our account.

```bash --move-call $PACKAGE_ID::todo_list::new \

## assigns the result of the "new" function to the "list" variable (from the previous step)

--assign list \

## transfers the object to the sender

--transfer-objects "[list]" sender ```

Once the command is constructed, you can run it in the terminal. If everything is correct, you should see the output similar to the one we had in previous sections. The output will contain the transaction digest, the transaction data, and the transaction effects.

```bash Transaction Digest: BJwYEnuuMzU4Y8cTwMoJbbQA6cLwPmwxvsRpSmvThoK8

~~~
╭──────────────────────────────────────────────────────────────────────────────────────────╮
│ Transaction Data │
├──────────────────────────────────────────────────────────────────────────────────────────┤
│ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Owner: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas
Budget: 100000000 MIST │ │ Gas Price: 1000 MIST │ │ Gas Payment: │ │ ┌── │ │ │ ID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Version: 22
│ │ │ Digest: DiBrBMshDiD9cThpaEgpcYSF76uV4hCoE1qRyQ3rnYCB │ │ └── │ │ │ │ Transaction Kind: Programmable │ │
╭────────────────────────────────────────────────────────────────────────────────────────╮ │ │
│ Input Objects │ │ │
├────────────────────────────────────────────────────────────────────────────────────────┤ │ │
│ 0 Pure Arg: Type: address, Value: "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" │ │ │
╰────────────────────────────────────────────────────────────────────────────────────────╯ │ │
╭──────────────────────────────────────────────────────────────────────────╮ │ │ │ Commands │ │ │
├──────────────────────────────────────────────────────────────────────────┤ │ │ │ 0 MoveCall: │ │ │ │ ┌ │ │ │ │ │
Function: new │ │ │ │ │ Module: todo_list │ │ │ │ │ Package: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ │ └ │ │ │ │ │ │ │ │ 1 TransferObjects:
│ │ │ │ ┌ │ │ │ │ │ Arguments: │ │ │ │ │ Result 0 │ │ │ │ │ Address: Input 0 │ │ │ │ └ │ │ │
╰──────────────────────────────────────────────────────────────────────────╯ │ │ │ │ Signatures: │ │
C5Lie4dtP5d3OkKzFBa+xM0BiNoB/A4ItthDCRTRBUrEE+jXeNs7mP4AuGwi3nzfTskh29+R1j1Kba4Wdy3QDA== │ │ │
╰──────────────────────────────────────────────────────────────────────────────────────────╯

╭──────────────────────────────────────────────────────────────────────────────────────╮ │ Transaction
Effects │ ├──────────────────────────────────────────────────────────────────────────────────────┤ │
Digest: BJwYEnuuMzU4Y8cTwMoJbbQA6cLwPmwxvsRpSmvThoK8 │ │ Status: Success │ │ Executed Epoch: 1213 │ │ │ │ Created Objects: │ │ ┌── │ │ │ ID:
0x74973c4ea2e78dc409f60481e23761cee68a48156df93a93fbcceb77d1cacdf6 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │
│ │ Version: 23 │ │ │ Digest: DuHTozDHMsuA7cFnWRQ1Gb8FQghAEBaj3inasJxqYq1c │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 )
│ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │ │ Gas Object: │ │ ┌── │ │ │ ID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 )
│ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │ │ Gas Cost Summary: │ │ Storage Cost: 2318000 MIST │ │ Computation Cost:
1000000 MIST │ │ Storage Rebate: 978120 MIST │ │ Non-refundable Storage Fee: 9880 MIST │ │ │ │ Transaction Dependencies: │ │
FSz2fYXmKqTf77mFXNq5JK7cKY8agWja7V5yDKEgL8c3 │ │ GgMZKTt482DYApbAZkPDtdssGHZLbxgjm2uMXhzJax8Q │
╰──────────────────────────────────────────────────────────────────────────────────────╯

╭──────────────────────────────╮ │ No transaction block events │ ╰──────────────────────────────╯

╭──────────────────────────────────────────────────────────────────────────────────────╮ │ Object
Changes │
├──────────────────────────────────────────────────────────────────────────────────────┤ │ Created
Objects: │ │ ┌── │ │ │ ObjectID: 0x74973c4ea2e78dc409f60481e23761cee68a48156df93a93fbcceb77d1cacdf6 │ │ │ Sender:
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │
│ │ ObjectType: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ Version: 23 │ │ │ Digest:
DuHTozDHMsuA7cFnWRQ1Gb8FQghAEBaj3inasJxqYq1c │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ Owner:
Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │ Version: 23 │ │ │ Digest:
82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │
╰──────────────────────────────────────────────────────────────────────────────────────╯

╭──────────────────────────────────────────────────────────────────────────────────────╮ │ Balance
Changes │ ├──────────────────────────────────────────────────────────────────────────────────────┤ │
┌── │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ CoinType: 0x2::sui::SUI │ │ │ Amount: -2339880 │ │ └── │
╰──────────────────────────────────────────────────────────────────────────────────────╯ ```
```

The section that we want to focus on is the "Object Changes". More specifically, the "Created Objects" part of it. It contains the object ID, the type and the version of the TodoList that you have created. We will use this object ID to interact with the list.

```
bash ╭──────────────────────────────────────────────────────────────────────────╮ │ Object Changes │
├──────────────────────────────────────────────────────────────────────────┤ │ Created Objects: │ │ ┌── │ │ │ ObjectID:
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType:
0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ Version: 22 │ │ │ Digest:
HyWdUpjuhjLY38dLpg6KPHQ3bt4BqQAbdF5gB8HQdEqG │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │
Version: 22 │ │ │ Digest: DiBrBMshDiD9cThpaEgpcYSF76uV4hCoE1qRyQ3rnYCB │ │ └── │
╰──────────────────────────────────────────────────────────────────────────╯
```
```

In this example the object ID is 0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 . And the owner should be your account address. We achieved this by transferring the object to the sender in the last command of the transaction.

Another way to test that you have successfully created the list is to check the account objects.

```
bash $ sui client objects
```

It should have an object that looks similar to this:

```
bash ╭ ... ╮ │ ╭──────────────┬──────────────────────────────────────────────────────────────────────────────╮ │ │ │ objectId │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ │ version │ 22 │ │ │ │ digest │ /DUEiCLkaNSgzpZSq2vSV0auQQEQhyH9occq9grMBZM= │ │
│ │ objectType │ 0x468d..29fe::todo_list::TodoList │ │ │ ╰──────────────┴──────────────────────────────────────────────────────────────────────────────╯ │ │ ... │
```

The TodoList that we created in the previous step is an object that you can interact with as its owner. You can call functions defined in the todo_list module on this object. To demonstrate this, we will add an item to the list. First, we will add just one item, and in the second transaction we will add 3 and remove another one.

Double check that you have variables set up from the previous step , and then add one more variable for the list object.

```
bash $ export LIST_ID=0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553
```

Now we can construct the transaction to add an item to the list. The command will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Finish the Hello, Sui chapter'"
```

In this command, we are calling the add function in the todo_list package. The function takes two arguments: the list object and the item to add. The item is a string, so we need to wrap it in single quotes. The command will add the item to the list.

If everything is correct, you should see the output similar to the one we had in previous sections. Now you can check the list object to see if the item was added.

```
bash $ sui client object $LIST_ID
```

The output should contain the item that you have added.

```
bash ╭──────────────┬──────────────────────────────────────────────────────────────────────────────────────────────────╮ │ objectId │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ version │ 24 │ │ digest │ FGcXH8MGpMs5BdTnC62CQ3VLAwwexYg2id5DKU7Jr9aQ │ │ objType
│ 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ owner │
╭─────────────┬───────────────────────────────────────────────────────────────────────╮ │ │ │ │ AddressOwner │
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ │
╰─────────────┴───────────────────────────────────────────────────────────────────────╯ │ │ prevTx │ EJVK6FEHtfTdCuGkNsU1HcrmUBEN6H6jshfcptnw8Yt1 │ │
storageRebate │ 1558000 │ │ content │ ╭────────────────────┬─────────────────────────────────────────────────────────────────────────────╮
│ │ │ │ dataType │ moveObject │ │ │ │ │ type │ 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ │ │
hasPublicTransfer │ true │ │ │ │ │ fields │ ╭─────────┬──────────────────────────────────────────────────────────────────────╮ │ │ │ │ │ │ │ id │
╭────┬──────────────────────────────────────────────────────────────────────╮ │ │ │ │ │ │ │ │ │ │ id │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ │ │ │ │ │ │ │
╰────┴──────────────────────────────────────────────────────────────────────╯ │ │ │ │ │ │ │ │ items │ ╭───────────────────────────────────╮ │ │ │ │ │ │ │
│ │ │ finish the Hello, Sui chapter │ │ │ │ │ │ │ │ │ │ ╰───────────────────────────────────╯
╰────────┴──────────────────────────────────────────────────────────────────────────╯ │ │ │ │
╰─────────────────────┴──────────────────────────────────────────────────────────────────────────────────────╯ │
╰───────────────┴───────────────────────────────────────────────────────────────────────────────────────────────────────╯
```

A JSON representation of the object can be obtained by adding the --json flag to the command.

```
bash $ sui client object $LIST_ID --json
```

```
bash { "objectId": "0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553", "version": "24", "digest":
"FGcXH8MGpMs5BdTnC62CQ3VLAwwexYg2id5DKU7Jr9aQ", "type": "0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList",
"owner": { "AddressOwner": "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" }, "previousTransaction":
"EJVK6FEHtfTdCuGkNsU1HcrmUBEN6H6jshfcptnw8Yt1", "storageRebate": "1558000", "content": { "dataType": "moveObject", "type":
"0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList", "hasPublicTransfer": true, "fields": { "id": { "id":
"0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553" }, "items": ["Finish the Hello, Sui chapter"] } } }
```

You can chain multiple commands in a single transaction. This shows the power of Transaction Blocks! Using the same list object, we will add three more items and remove one. The command will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Finish Concepts chapter'" \ --move-call
$PACKAGE_ID::todo_list::add @$LIST_ID "'Read the Move Basics chapter'" \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Learn about Object Model'" \
--move-call $PACKAGE_ID::todo_list::remove @$LIST_ID 0
```

If previous commands were successful, this one should not be any different. You can check the list object to see if the items were added and removed. The JSON representation is a bit more readable!

```
bash sui client object $LIST_ID --json
```

```
bash { "objectId": "0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553", "version": "25", "digest":
"EDTXDsteqPGAGu4zFAj5bbQGTkucWk4hhuUquk39enGA", "type": "0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList",
"owner": { "AddressOwner": "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" }, "previousTransaction":
"7SXLGBSh31jv8G7okQ9mEgnw5MnTfvzzHEHpWf3Sa9gY", "storageRebate": "1922800", "content": { "dataType": "moveObject", "type":
"0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList", "hasPublicTransfer": true, "fields": { "id": { "id":
"0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553" }, "items": [ "Finish Concepts chapter", "Read the Move Basics chapter", "Learn
about Object Model" ] } } }
```

Commands don't have to be in the same package or operate on the same object. Within a single transaction block, you can interact with multiple packages and objects. This is a powerful feature that allows you to build complex interactions on-chain!

In this guide, we have shown how to publish a package on the Move blockchain and interact with it using the Sui CLI. We have demonstrated how to create a new list object, add items to it, and remove them. We have also shown how to chain multiple commands in a single transaction block. This guide should give you a good starting point for building your own applications on the Sui blockchain!

## Build the package

To make sure that we did everything correctly, let's build the package by running the sui move build command. If everything is correct, you should see the output similar to the following:

```
bash $ sui move build UPDATING GIT DEPENDENCY https://github.com/MystenLabs/sui.git INCLUDING DEPENDENCY Bridge INCLUDING DEPENDENCY DeepBook INCLUDING
DEPENDENCY SuiSystem INCLUDING DEPENDENCY Sui INCLUDING DEPENDENCY MoveStdlib BUILDING todo_list
```

If there are no errors following this output, you have successfully built the package. If there are errors, make sure that:

There are not many other reasons for the code to fail at this stage. But if you are still having issues, try looking up the structure of the package in this location .

If you already have an account set up, you can skip this step.

To publish and interact with the package, we need to set up an account. While developing, the best option for doing so is to run your own Local Network . For now you just need to run RUST_LOG="off,sui_node=info" sui start --with-faucet --force-regenesis . The Sui Local Network will run on port 9000 of your machine, so make sure that the port isn't being used by any other application.

If you are doing it for the first time, you will need to create a new account. To do this, run the sui client command, then the CLI will prompt you with multiple questions. The answers are marked below with > :

```bash $ sui client Config file ["/path/to/home/.sui/sui_config/client.yaml"] doesn't exist, do you want to connect to a Sui Full node server [y/N]?
```

y Sui Full node server URL (Defaults to Sui Testnet if not specified) : http://127.0.0.1:9000 Environment alias for [http://127.0.0.1:9000] : localnet Select key scheme to generate keypair (0 for ed25519, 1 for secp256k1, 2: for secp256r1): 0 ```

After you have answered the questions, the CLI will generate a new keypair and save it to the configuration file. You can now use this account to interact with the network.

To check that we have the account set up correctly, run the sui client active-address command:

```
bash $ sui client active-address 0x....
```

The command will output the address of your account, it starts with 0x followed by 64 characters.

In devnet and testnet environments, the CLI provides a way to request coins to your account, so you can interact with the network. To request coins, run the sui client faucet command:

```
bash $ sui client faucet Request successful. It can take up to 1 minute to get the coin. Run sui client gas to check your gas coins.
```

After waiting a little bit, you can check that the Coin object was sent to your account by running the sui client balance command:

```
bash $ sui client balance ╭────────────────────────────────────────╮ │ Balance of coins owned by this address │
├────────────────────────────────────────┤ │ ╭──────────────────────────────────────╮ │ │ │ coin balance (raw) balance │ │ │
├──────────────────────────────────────┤ │ │ │ Sui 1000000000 1.00 SUI │ │ │ ╰──────────────────────────────────────╯ │
╰────────────────────────────────────────╯
```

Alternatively, you can query objects owned by your account, by running the sui client objects command. The actual output will be different, because the object ID is unique, and so is digest, but the structure will be similar:

```
bash $ sui client objects ╭───────────────────────────────────────────────────────────────────────────────────────╮ │
╭────────────┬──────────────────────────────────────────────────────────────────────╮ │ │ │ objectId │
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ │ version │ 4 │ │ │ │ digest │ nA68oa8gab/CdIRw+240wze8u0P+sRe4vcisbENcR4U= │ │
│ │ objectType │ 0x0000..0002::coin::Coin │ │ │ ╰────────────┴──────────────────────────────────────────────────────────────────────╯ │
╰───────────────────────────────────────────────────────────────────────────────────────╯
```

Now that we have the account set up and the coins in the account, we can interact with the network. We will start by publishing the package to the network.

To publish the package to the network, we will use the sui client publish command. The command will automatically build the package and use its bytecode to publish in a single transaction.

We are using the --gas-budget argument during publishing. It specifies how much gas we are willing to spend on the transaction. We won't touch on this topic in this section, but it's important to know that every transaction in Sui costs gas, and the gas is paid in SUI coins.

The gas-budget is specified in MISTs . 1 SUI equals 10^9 MISTs. For the sake of demonstration, we will use 100,000,000 MISTs, which is 0.1 SUI.

```bash

# run this from the `todo_list` folder

$ sui client publish --gas-budget 100000000

# alternatively, you can specify path to the package

$ sui client publish --gas-budget 100000000 todo_list ```

The output of the publish command is rather lengthy, so we will show and explain it in parts.

```
bash $ sui client publish --gas-budget 100000000 UPDATING GIT DEPENDENCY https://github.com/MystenLabs/sui.git INCLUDING DEPENDENCY Bridge INCLUDING
DEPENDENCY DeepBook INCLUDING DEPENDENCY SuiSystem INCLUDING DEPENDENCY Sui INCLUDING DEPENDENCY MoveStdlib BUILDING todo_list Successfully verified
dependencies on-chain against source. Transaction Digest: GpcDV6JjjGQMRwHpEz582qsd5MpCYgSwrDAq1JXcpFjW
```

As you can see, when we run the publish command, the CLI first builds the package, then verifies the dependencies on-chain, and finally publishes the package. The output of the command is the transaction digest, which is a unique identifier of the transaction and can be used to query the transaction status.

The section titled TransactionData contains the information about the transaction we just sent. It features fields like sender , which is your address, the gas_budget set with the --gas-budget argument, and the Coin we used for payment. It also prints the Commands that were run by the CLI. In this example, the commands Publish and TransferObject were run - the latter transfers a special object UpgradeCap to the sender.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────────╮ │ Transaction Data │
├──────────────────────────────────────────────────────────────────────────────────────────┤ │ Sender:
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Owner: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │
Gas Budget: 100000000 MIST │ │ Gas Price: 1000 MIST │ │ Gas Payment: │ │ ┌── │ │ │ ID: 0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de
│ │ │ Version: 7 │ │ │ Digest: AXYPnups8A5J6pkvLa6RekX2ye3qur66EZ88mEbaUDQ1 │ │ └── │ │ │ │ Transaction Kind: Programmable │ │
╭────────────────────────────────────────────────────────────────────────────────────╮ │ │ │ Input Objects │ │ │
├────────────────────────────────────────────────────────────────────────────────────┤ │ │ │ 0 Pure Arg: Type: address, Value:
"0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" │ │ │
╰────────────────────────────────────────────────────────────────────────────────────╯ │ │
╭──────────────────────────────────────────────────────────────────╮ │ │ │ Commands │ │ │
├──────────────────────────────────────────────────────────────────┤ │ │ │ 0 Publish: │ │ │ │ ┌ │ │ │ │ │ Dependencies: │ │ │ │ │
0x0000000000000000000000000000000000000000000000000000000000000001 │ │ │ │ │ 0x0000000000000000000000000000000000000000000000000000000000000002 │ │ │ │ └
│ │ │ │ │ │ │ │ 1 TransferObjects: │ │ │ │ ┌ │ │ │ │ │ Arguments: │ │ │ │ │ Result 0 │ │ │ │ │ Address: Input 0 │ │ │ │ └ │ │ │
╰──────────────────────────────────────────────────────────────────╯ │ │ │ │ Signatures: │ │
gebjSbVwZwTkizfYg2XIuzdx+d66VxFz8EmVaisVFiV3GkDay6L+hQG3n2CQ1hrWphP6ZLc7bd1WRq4ss+hQAQ== │ │ │
╰──────────────────────────────────────────────────────────────────────────────────────────╯
```

Transaction Effects contains the status of the transaction, the changes that the transaction made to the state of the network and the objects involved in the transaction.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────╮ │ Transaction Effects │
├──────────────────────────────────────────────────────────────────────────────────────┤ │ Digest:
GpcDV6JjjGQMRwHpEz582qsd5MpCYgSwrDAq1JXcpFjW │ │ Status: Success │ │ Executed Epoch: 411 │ │ │ │ Created Objects: │ │ ┌── │ │ │ ID:
0x160f7856e13b27e5a025112f361370f4efc2c2659cb0023f1e99a8a84d1652f3 │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 8y6bhwvQrGJHDckUZmj2HDAjfkyVqHohhvY1Fvzyj7ec │ │ └──
│ │ ┌── │ │ │ ID: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ Owner: Immutable │ │ │ Version: 1 │ │ │ Digest:
Ein91NF2hc3qC4XYoMUFMfin9U23xQmDAdEMSHLae7MK │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ID:
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └──
│ │ Gas Object: │ │ ┌── │ │ │ ID: 0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └──
│ │ Gas Cost Summary: │ │ Storage Cost: 10404400 MIST │ │ Computation Cost: 1000000 MIST │ │ Storage Rebate: 978120 MIST │ │ Non-refundable Storage Fee:
9880 MIST │ │ │ │ Transaction Dependencies: │ │ 7Ukrc5GqdFqTA41wvWgreCdHn2vRLfgQ3YMFkdks72Vk │ │ 7d4amuHGhjtYKujEs9YkJARzNEn4mRbWWv3fn4cdKdyh │
╰──────────────────────────────────────────────────────────────────────────────────────╯
```

If there were any events emitted, you would see them in this section. Our package does not use events, so the section is empty.

```
bash ╭─────────────────────────────╮ │ No transaction block events │ ╰─────────────────────────────╯
```

These are the changes to objects that transaction has made. In our case, we have created a new UpgradeCap object which is a special object that allows the sender to upgrade the package in the future, mutated the Gas object, and published a new package. Packages are also objects on Sui.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────╮ │ Object Changes │
├──────────────────────────────────────────────────────────────────────────────────────┤ │ Created Objects: │ │ ┌── │ │ │ ObjectID:
```
```

```
0x160f7856e13b27e5a025112f361370f4efc2c2659cb0023f1e99a8a84d1652f3 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::package::UpgradeCap │ │ │ Version: 8
│ │ │ Digest: 8y6bhwvQrGJHDckUZmj2HDAjfkyVqHohhvY1Fvzyj7ec │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │
Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └── │ │ Published Objects: │ │ ┌── │ │ │ PackageID:
0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ Version: 1 │ │ │ Digest: Ein91NF2hc3qC4XYoMUFMfin9U23xQmDAdEMSHLae7MK │ │ │
Modules: todo_list │ │ └── │ ╰──────────────────────────────────────────────────────────────────────╯
```

This last section contains changes to SUI Coins, in our case, we have spent around 0.015 SUI, which in MIST is 10,500,000. You can see it under the amount field in the output.

```
bash ╭──────────────────────────────────────────────────────────────────────────────╮ │ Balance Changes │
├──────────────────────────────────────────────────────────────────────────────┤ │ ┌── │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ CoinType: 0x2::sui::SUI │ │ │ Amount: -10426280 │ │ └── │
╰──────────────────────────────────────────────────────────────────────────────╯
```

It is possible to specify the --json flag during publishing to get the output in JSON format. This is useful if you want to parse the output programmatically or store it for later use.

```
bash $ sui client publish --gas-budget 100000000 --json
```

After the package is published on chain, we can interact with it. To do this, we need to find the address (object ID) of the package. It's under the Published Objects section of the Object Changes output. The address is unique for each package, so you will need to copy it from the output.

In this example, the address is:

```
bash 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe
```

Now that we have the address, we can interact with the package. In the next section, we will show how to interact with the package by sending transactions.

To demonstrate the interaction with the todo_list package, we will send a transaction to create a new list and add an item to it. Transactions are sent via the sui client ptb command, it allows using the Transaction Blocks at full capacity. The command may look big and complex, but we go through it step by step.

Before we construct the command, let's store the values we will use in the transaction. Replace the 0x4.... with the address of the package you have published. And MY_ADDRESS variable will be automatically set to your address from the CLI output.

```
bash $ export PACKAGE_ID=0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe $ export MY_ADDRESS=$(sui client active-address)
```

Now to building an actual transaction. The transaction will consist of two parts: we will call the new function in the todo_list package to create a new list, and then we will transfer the list object to our account. The transaction will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --assign sender @$MY_ADDRESS \ --move-call $PACKAGE_ID::todo_list::new \ --assign list \ --transfer-
objects "[list]" sender
```

In this command, we are using the ptb subcommand to build a transaction. Parameters that follow it define the actual commands and actions that the transaction will perform. The first two calls we make are utility calls to set the sender address to the command inputs and set the gas budget for the transaction.

~~~bash

## sets the gas budget for the transaction

--gas-budget 100000000 \n

## registers a variable "sender=@..."

--assign sender @$MY_ADDRESS \n ```

Then we perform the actual call to a function in the package. We use the --move-call followed by the package ID, the module name, and the function name. In this case, we are calling the new function in the todo_list package.

```bash

## calls the "new" function in the "todo_list" package under the $PACKAGE_ID address

--move-call $PACKAGE_ID::todo_list::new ```

The function that we defined actually returns a value, which we want need to store. We use the --assign command to give a name to the returned value. In this case, we are calling it list . And then we transfer the object to our account.

```bash --move-call $PACKAGE_ID::todo_list::new \

## assigns the result of the "new" function to the "list" variable (from the previous step)

--assign list \

## transfers the object to the sender

--transfer-objects "[list]" sender ```

Once the command is constructed, you can run it in the terminal. If everything is correct, you should see the output similar to the one we had in previous sections. The output will contain the transaction digest, the transaction data, and the transaction effects.

```bash Transaction Digest: BJwYEnuuMzU4Y8cTwMoJbbQA6cLwPmwxvsRpSmvThoK8

╭──────────────────────────────────────────────────────────────────────╮

│ Transaction Data │

├──────────────────────────────────────────────────────────────────────┤

│ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Owner: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Budget: 100000000 MIST │ │ Gas Price: 1000 MIST │ │ Gas Payment: │ │ ┌── │ │ │ ID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Version: 22 │ │ │ Digest: DiBrBMshDiD9cThpaEgpcYSF76uV4hCoE1qRyQ3rnYCB │ │ └── │ │ │ │ Transaction Kind: Programmable │ │

╭──────────────────────────────────────────────────────────────────╮ │ │

│ Input Objects │ │ │

├──────────────────────────────────────────────────────────────────┤ │ │

│ 0 Pure Arg: Type: address, Value: "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" │ │ │

╰──────────────────────────────────────────────────────────────────╯ │ │

╭──────────────────────────────────────────────────────╮ │ │ │ Commands │ │ │
├──────────────────────────────────────────────────────┤ │ │ │ 0 MoveCall: │ │ │ │ ┌ │ │ │ │ │
Function: new │ │ │ │ │ Module: todo_list │ │ │ │ │ Package: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ │ └ │ │ │ │ │ │ │ │ 1 TransferObjects:
~~~

```
│ │ │ │ ┌ │ │ │ │ │ Arguments: │ │ │ │ │ Result 0 │ │ │ │ │ Address: Input 0 │ │ │ │ └ │ │ │
╰──────────────────────────────────────────────────────────────────────╯ │ │ │ │ Signatures: │ │
C5Lie4dtP5d3OkKzFBa+xM0BiNoB/A4ItthDCRTRBUrEE+jXeNs7mP4AuGwi3nzfTskh29+R1j1Kba4Wdy3QDA== │ │ │
╰──────────────────────────────────────────────────────────────────────────────╯

╭──────────────────────────────────────────────────────────────────────╮ │ Transaction
Effects │ ├──────────────────────────────────────────────────────────────────┤ │
Digest: BJwYEnuuMzU4Y8cTwMoJbbQA6cLwPmwxvsRpSmvThoK8 │ │ Status: Success │ │ Executed Epoch: 1213 │ │ │ │ Created Objects: │ │ ┌── │ │ │ ID:
0x74973c4ea2e78dc409f60481e23761cee68a48156df93a93fbcceb77d1cacdf6 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │
│ │ Version: 23 │ │ │ Digest: DuHTozDHMsuA7cFnWRQ1Gb8FQghAEBaj3inasJxqYq1c │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 )
│ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │ │ Gas Object: │ │ ┌── │ │ │ ID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 )
│ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │ │ Gas Cost Summary: │ │ Storage Cost: 2318000 MIST │ │ Computation Cost:
1000000 MIST │ │ Storage Rebate: 978120 MIST │ │ Non-refundable Storage Fee: 9880 MIST │ │ │ │ Transaction Dependencies: │ │
FSz2fYXmKqTf77mFXNq5JK7cKY8agWja7V5yDKEgL8c3 │ │ GgMZKTt482DYApbAZkPDtdssGHZLbxgjm2uMXhzJax8Q │
╰──────────────────────────────────────────────────────────────────────╯

╭──────────────────────────────╮ │ No transaction block events │ ╰──────────────────────────────╯

╭──────────────────────────────────────────────────────────────────────────╮ │ Object
Changes │
├──────────────────────────────────────────────────────────────────────────┤ │ Created
Objects: │ │ ┌── │ │ │ ObjectID: 0x74973c4ea2e78dc409f60481e23761cee68a48156df93a93fbcceb77d1cacdf6 │ │ │ Sender:
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │
│ │ ObjectType: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ Version: 23 │ │ │ Digest:
DuHTozDHMsuA7cFnWRQ1Gb8FQghAEBaj3inasJxqYq1c │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ Owner:
Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │ Version: 23 │ │ │ Digest:
82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │
╰──────────────────────────────────────────────────────────────────────────╯

╭──────────────────────────────────────────────────────────────────────╮ │ Balance
Changes │ ├──────────────────────────────────────────────────────────────────────┤ │
┌── │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ CoinType: 0x2::sui::SUI │ │ │ Amount: -2339880 │ │ └── │
╰──────────────────────────────────────────────────────────────────────╯ ```
```

The section that we want to focus on is the "Object Changes". More specifically, the "Created Objects" part of it. It contains the object ID, the type and the version of the TodoList that you have created. We will use this object ID to interact with the list.

```
bash ╭──────────────────────────────────────────────────────────────╮ │ Object Changes │
├──────────────────────────────────────────────────────────────┤ │ Created Objects: │ │ ┌── │ │ │ ObjectID:
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType:
0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ Version: 22 │ │ │ Digest:
HyWdUpjuhjLY38dLpg6KPHQ3bt4BqQAbdF5gB8HQdEqG │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │
Version: 22 │ │ │ Digest: DiBrBMshDiD9cThpaEgpcYSF76uV4hCoE1qRyQ3rnYCB │ │ └── │
╰──────────────────────────────────────────────────────────────╯
```

In this example the object ID is 0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 . And the owner should be your account address. We achieved this by transferring the object to the sender in the last command of the transaction.

Another way to test that you have successfully created the list is to check the account objects.

```
bash $ sui client objects
```

It should have an object that looks similar to this:

```
bash ╭ ... ╮ │ ╭──────────┬──────────────────────────────────────────────╮ │ │ │ objectId │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ │ version │ 22 │ │ │ │ digest │ /DUEiCLkaNSgzpZSq2vSV0auQQEQhyH9occq9grMBZM= │ │
│ │ objectType │ 0x468d..29fe::todo_list::TodoList │ │ │ ╰──────────┴──────────────────────────────────────────────╯ │ │ ... │
```

The TodoList that we created in the previous step is an object that you can interact with as its owner. You can call functions defined in the todo_list module on this object. To demonstrate this, we will add an item to the list. First, we will add just one item, and in the second transaction we will add 3 and remove another one.

Double check that you have variables set up [from the previous step](), and then add one more variable for the list object.

```
bash $ export LIST_ID=0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553
```

Now we can construct the transaction to add an item to the list. The command will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Finish the Hello, Sui chapter'"
```

In this command, we are calling the add function in the todo_list package. The function takes two arguments: the list object and the item to add. The item is a string, so we need to wrap it in single quotes. The command will add the item to the list.

If everything is correct, you should see the output similar to the one we had in previous sections. Now you can check the list object to see if the item was added.

```
bash $ sui client object $LIST_ID
```

The output should contain the item that you have added.

```
bash ╭──────────┬──────────────────────────────────────────────────────╮ │ objectId │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ version │ 24 │ │ digest │ FGcXH8MGpMs5BdTnC62CQ3VLAwwexYg2id5DKU7Jr9aQ │ │ objType
│ 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ owner │
╭────────┬────────────────────────────────────────────╮ │ │ │ │ AddressOwner │
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ │
╰────────┴────────────────────────────────────────────╯ │ │ prevTx │ EJVK6FEHtfTdCuGkNsU1HcrmUBEN6H6jshfcptnw8Yt1 │ │
storageRebate │ 1558000 │ │ content │ ╭──────────┬──────────────────────────────────╮
│ │ │ │ dataType │ moveObject │ │ │ │ │ type │ 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ │ │
hasPublicTransfer │ true │ │ │ │ │ fields │ ╭───────┬──────────────────────────╮ │ │ │ │ │ │ │ id │
╭───┬──────────────────────────────────────────────╮ │ │ │ │ │ │ │ │ │ │ id │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ │ │ │ │ │ │ │
╰───┴──────────────────────────────────────────────╯ │ │ │ │ │ │ │ │ items │ ╭──────────────────────╮ │ │ │ │ │ │ │
│ │ │ finish the Hello, Sui chapter │ │ │ │ │ │ │ │ │ │ ╰──────────────────────╯ │ │ │ │ │ │ │
╰───┴──────────────────────────────────────────────╯ │ │ │ │
╰──────────┴──────────────────────────────────────────╯ │
╰──────────┴──────────────────────────────────────────────────────╯
```

A JSON representation of the object can be obtained by adding the --json flag to the command.

```
bash $ sui client object $LIST_ID --json
```

```
bash { "objectId": "0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553", "version": "24", "digest":
"FGcXH8MGpMs5BdTnC62CQ3VLAwwexYg2id5DKU7Jr9aQ", "type": "0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList",
"owner": { "AddressOwner": "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" }, "previousTransaction":
```

```
"EJVK6FEHtfTdCuGkNsU1HcrmUBEN6H6jshfcptnw8Yt1", "storageRebate": "1558000", "content": { "dataType": "moveObject", "type":
"0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList", "hasPublicTransfer": true, "fields": { "id": { "id":
"0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553" }, "items": ["Finish the Hello, Sui chapter"] } } }
```

You can chain multiple commands in a single transaction. This shows the power of Transaction Blocks! Using the same list object, we will add three more items and remove one. The command will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Finish Concepts chapter'" \ --move-call
$PACKAGE_ID::todo_list::add @$LIST_ID "'Read the Move Basics chapter'" \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Learn about Object Model'" \
--move-call $PACKAGE_ID::todo_list::remove @$LIST_ID 0
```

If previous commands were successful, this one should not be any different. You can check the list object to see if the items were added and removed. The JSON representation is a bit more readable!

```
bash sui client object $LIST_ID --json
```

```
bash { "objectId": "0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553", "version": "25", "digest":
"EDTXDsteqPGAGu4zFAj5bbQGTkucWk4hhuUquk39enGA", "type": "0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList",
"owner": { "AddressOwner": "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" }, "previousTransaction":
"7SXLGBSh31jv8G7okQ9mEgnw5MnTfvzzHEHpWf3Sa9gY", "storageRebate": "1922800", "content": { "dataType": "moveObject", "type":
"0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList", "hasPublicTransfer": true, "fields": { "id": { "id":
"0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553" }, "items": [ "Finish Concepts chapter", "Read the Move Basics chapter", "Learn
about Object Model" ] } } }
```

Commands don't have to be in the same package or operate on the same object. Within a single transaction block, you can interact with multiple packages and objects. This is a powerful feature that allows you to build complex interactions on-chain!

In this guide, we have shown how to publish a package on the Move blockchain and interact with it using the Sui CLI. We have demonstrated how to create a new list object, add items to it, and remove them. We have also shown how to chain multiple commands in a single transaction block. This guide should give you a good starting point for building your own applications on the Sui blockchain!

## Set up an account

If you already have an account set up, you can skip this step.

To publish and interact with the package, we need to set up an account. While developing, the best option for doing so is to run your own Local Network . For now you just need to run RUST_LOG="off,sui_node=info" sui start --with-faucet --force-regenesis . The Sui Local Network will run on port 9000 of your machine, so make sure that the port isn't being used by any other application.

If you are doing it for the first time, you will need to create a new account. To do this, run the sui client command, then the CLI will prompt you with multiple questions. The answers are marked below with > :

```bash $ sui client Config file ["/path/to/home/.sui/sui_config/client.yaml"] doesn't exist, do you want to connect to a Sui Full node server [y/N]?

> y Sui Full node server URL (Defaults to Sui Testnet if not specified) : http://127.0.0.1:9000 Environment alias for [http://127.0.0.1:9000] : localnet Select key scheme to generate keypair (0 for ed25519, 1 for secp256k1, 2: for secp256r1): 0 ```

After you have answered the questions, the CLI will generate a new keypair and save it to the configuration file. You can now use this account to interact with the network.

To check that we have the account set up correctly, run the sui client active-address command:

```
bash $ sui client active-address 0x....
```

The command will output the address of your account, it starts with 0x followed by 64 characters.

In devnet and testnet environments, the CLI provides a way to request coins to your account, so you can interact with the network. To request coins, run the sui client faucet command:

```
bash $ sui client faucet Request successful. It can take up to 1 minute to get the coin. Run sui client gas to check your gas coins.
```

After waiting a little bit, you can check that the Coin object was sent to your account by running the sui client balance command:

```
bash $ sui client balance ╭────────────────────────────────────────╮ │ Balance of coins owned by this address │
├────────────────────────────────────────┤ │ ╭──────────────────────────────────────╮ │ │ │ coin balance (raw) balance │ │ │
├──────────────────────────────────────┤ │ │ │ Sui 1000000000 1.00 SUI │ │ │ ╰──────────────────────────────────────╯ │
╰────────────────────────────────────────╯
```

Alternatively, you can query objects owned by your account, by running the sui client objects command. The actual output will be different, because the object ID is unique, and so is digest, but the structure will be similar:

```
bash $ sui client objects ╭───────────────────────────────────────────────────────────────────────────────────────────╮ │
╭────────────┬──────────────────────────────────────────────────────────────────────╮ │ │ │ objectId │
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ │ version │ 4 │ │ │ │ digest │ nA68oa8gab/CdIRw+240wze8u0P+sRe4vcisbENcR4U= │ │
│ │ objectType │ 0x0000..0002::coin::Coin │ │ │ ╰────────────┴──────────────────────────────────────────────────────────────────────╯ │
╰───────────────────────────────────────────────────────────────────────────────────────────╯
~~~

Now that we have the account set up and the coins in the account, we can interact with the network. We will start by publishing the package to the network.

To publish the package to the network, we will use the sui client publish command. The command will automatically build the package and use its bytecode to publish in a single transaction.

We are using the --gas-budget argument during publishing. It specifies how much gas we are willing to spend on the transaction. We won't touch on this topic in this section, but it's important to know that every transaction in Sui costs gas, and the gas is paid in SUI coins.

The gas-budget is specified in MISTs . 1 SUI equals 10^9 MISTs. For the sake of demonstration, we will use 100,000,000 MISTs, which is 0.1 SUI.

```bash

## run this from the `todo_list` folder

$ sui client publish --gas-budget 100000000

## alternatively, you can specify path to the package

$ sui client publish --gas-budget 100000000 todo_list ```

The output of the publish command is rather lengthy, so we will show and explain it in parts.

~~~
bash $ sui client publish --gas-budget 100000000 UPDATING GIT DEPENDENCY https://github.com/MystenLabs/sui.git INCLUDING DEPENDENCY Bridge INCLUDING
DEPENDENCY DeepBook INCLUDING DEPENDENCY SuiSystem INCLUDING DEPENDENCY Sui INCLUDING DEPENDENCY MoveStdlib BUILDING todo_list Successfully verified
dependencies on-chain against source. Transaction Digest: GpcDV6JjjGQMRwHpEz582qsd5MpCYgSwrDAq1JXcpFjW
```

As you can see, when we run the publish command, the CLI first builds the package, then verifies the dependencies on-chain, and finally publishes the package. The output of the command is the transaction digest, which is a unique identifier of the transaction and can be used to query the transaction status.

The section titled TransactionData contains the information about the transaction we just sent. It features fields like sender , which is your address, the gas_budget set with the --gas-budget argument, and the Coin we used for payment. It also prints the Commands that were run by the CLI. In this example, the commands Publish and TransferObject were run - the latter transfers a special object
```

UpgradeCap to the sender.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────────────╮ │ Transaction Data │
├──────────────────────────────────────────────────────────────────────────────────────────────┤ │ Sender:
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Owner: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │
Gas Budget: 100000000 MIST │ │ Gas Price: 1000 MIST │ │ Gas Payment: │ │ ┌── │ │ │ ID: 0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de
│ │ │ Version: 7 │ │ │ Digest: AXYPnups8A5J6pkvLa6RekX2ye3qur66EZ88mEbaUDQ1 │ │ └── │ │ │ │ Transaction Kind: Programmable │ │
╭─────────────────────────────────────────────────────────────────────────────────────────────╮ │ │ │ Input Objects │ │ │
├─────────────────────────────────────────────────────────────────────────────────────────────┤ │ │ │ 0 Pure Arg: Type: address, Value:
"0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" │ │ │
╰─────────────────────────────────────────────────────────────────────────────────────────────╯ │ │
╭──────────────────────────────────────────────────────────────────╮ │ │ │ Commands │ │ │
├──────────────────────────────────────────────────────────────────┤ │ │ │ 0 Publish: │ │ │ │ ┌ │ │ │ │ │ Dependencies: │ │ │ │ │
0x0000000000000000000000000000000000000000000000000000000000000001 │ │ │ │ │ 0x0000000000000000000000000000000000000000000000000000000000000002 │ │ │ │ └
│ │ │ │ │ │ │ │ 1 TransferObjects: │ │ │ │ ┌ │ │ │ │ │ Arguments: │ │ │ │ │ Result 0 │ │ │ │ │ Address: Input 0 │ │ │ │ └ │ │ │
╰──────────────────────────────────────────────────────────────────╯ │ │ │ │ Signatures: │ │
gebjSbVwZwTkizfYg2XIuzdx+d66VxFz8EmVaisVFiV3GkDay6L+hQG3n2CQ1hrWphP6ZLc7bd1WRq4ss+hQAQ== │ │ │
╰──────────────────────────────────────────────────────────────────────────────────────────────╯
```

Transaction Effects contains the status of the transaction, the changes that the transaction made to the state of the network and the objects involved in the transaction.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────╮ │ Transaction Effects │
├──────────────────────────────────────────────────────────────────────────────────────┤ │ Digest:
GpcDV6JjjGQMRwHpEz582qsd5MpCYgSwrDAq1JXcpFjW │ │ Status: Success │ │ Executed Epoch: 411 │ │ │ │ Created Objects: │ │ ┌── │ │ │ ID:
0x160f7856e13b27e5a025112f361370f4efc2c2659cb0023f1e99a8a84d1652f3 │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 8y6bhwvQrGJHDckUZmj2HDAjfkyVqHohhvY1Fvzyj7ec │ │ └──
│ │ ┌── │ │ │ ID: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ Owner: Immutable │ │ │ Version: 1 │ │ │ Digest:
Ein91NF2hc3qC4XYoMUFMfin9U23xQmDAdEMSHLae7MK │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ID:
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └──
│ │ Gas Object: │ │ ┌── │ │ │ ID: 0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └──
│ │ Gas Cost Summary: │ │ Storage Cost: 10404400 MIST │ │ Computation Cost: 1000000 MIST │ │ Storage Rebate: 978120 MIST │ │ Non-refundable Storage Fee:
9880 MIST │ │ │ │ Transaction Dependencies: │ │ 7Ukrc5GqdFqTA41wvWgreCdHn2vRLfgQ3YMFkdks72Vk │ │ 7d4amuHGhjtYKujEs9YkJARzNEn4mRbWWv3fn4cdKdyh │
╰──────────────────────────────────────────────────────────────────────────────────────╯
```

If there were any events emitted, you would see them in this section. Our package does not use events, so the section is empty.

```
bash ╭─────────────────────────────╮ │ No transaction block events │ ╰─────────────────────────────╯
```

These are the changes to objects that transaction has made. In our case, we have created a new UpgradeCap object which is a special object that allows the sender to upgrade the package in the future, mutated the Gas object, and published a new package. Packages are also objects on Sui.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────╮ │ Object Changes │
├──────────────────────────────────────────────────────────────────────────────────────┤ │ Created Objects: │ │ ┌── │ │ │ ObjectID:
0x160f7856e13b27e5a025112f361370f4efc2c2659cb0023f1e99a8a84d1652f3 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::package::UpgradeCap │ │ │ Version: 8
│ │ │ Digest: 8y6bhwvQrGJHDckUZmj2HDAjfkyVqHohhvY1Fvzyj7ec │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │
Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └── │ │ Published Objects: │ │ ┌── │ │ │ PackageID:
0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ Version: 1 │ │ │ Digest: Ein91NF2hc3qC4XYoMUFMfin9U23xQmDAdEMSHLae7MK │ │ │
Modules: todo_list │ │ └── │ ╰──────────────────────────────────────────────────────────────────────────────────────╯
```

This last section contains changes to SUI Coins, in our case, we have spent around 0.015 SUI, which in MIST is 10,500,000. You can see it under the amount field in the output.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────╮ │ Balance Changes │
├──────────────────────────────────────────────────────────────────────────────────────┤ │ ┌── │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ CoinType: 0x2::sui::SUI │ │ │ Amount: -10426280 │ │ └── │
╰──────────────────────────────────────────────────────────────────────────────────────╯
```

It is possible to specify the --json flag during publishing to get the output in JSON format. This is useful if you want to parse the output programmatically or store it for later use.

```
bash $ sui client publish --gas-budget 100000000 --json
```

After the package is published on chain, we can interact with it. To do this, we need to find the address (object ID) of the package. It's under the Published Objects section of the Object Changes output. The address is unique for each package, so you will need to copy it from the output.

In this example, the address is:

```
bash 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe
```

Now that we have the address, we can interact with the package. In the next section, we will show how to interact with the package by sending transactions.

To demonstrate the interaction with the todo_list package, we will send a transaction to create a new list and add an item to it. Transactions are sent via the sui client ptb command, it allows using the Transaction Blocks at full capacity. The command may look big and complex, but we go through it step by step.

Before we construct the command, let's store the values we will use in the transaction. Replace the 0x4.... with the address of the package you have published. And MY_ADDRESS variable will be automatically set to your address from the CLI output.

```
bash $ export PACKAGE_ID=0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe $ export MY_ADDRESS=$(sui client active-address)
```

Now to building an actual transaction. The transaction will consist of two parts: we will call the new function in the todo_list package to create a new list, and then we will transfer the list object to our account. The transaction will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --assign sender @$MY_ADDRESS \ --move-call $PACKAGE_ID::todo_list::new \ --assign list \ --transfer-
objects "[list]" sender
```

In this command, we are using the ptb subcommand to build a transaction. Parameters that follow it define the actual commands and actions that the transaction will perform. The first two calls we make are utility calls to set the sender address to the command inputs and set the gas budget for the transaction.

~~~bash

# sets the gas budget for the transaction

--gas-budget 100000000 \n

# registers a variable "sender=@..."

--assign sender @$MY_ADDRESS \n ```

Then we perform the actual call to a function in the package. We use the --move-call followed by the package ID, the module name, and the function name. In this case, we are calling the new function in the todo_list package.

```bash
~~~

## calls the "new" function in the "todo_list" package under the $PACKAGE_ID address

--move-call $PACKAGE_ID::todo_list::new ```

The function that we defined actually returns a value, which we want need to store. We use the --assign command to give a name to the returned value. In this case, we are calling it list . And then we transfer the object to our account.

```bash --move-call $PACKAGE_ID::todo_list::new \

## assigns the result of the "new" function to the "list" variable (from the previous step)

--assign list \

## transfers the object to the sender

--transfer-objects "[list]" sender ```

Once the command is constructed, you can run it in the terminal. If everything is correct, you should see the output similar to the one we had in previous sections. The output will contain the transaction digest, the transaction data, and the transaction effects.

```bash Transaction Digest: BJwYEnuuMzU4Y8cTwMoJbbQA6cLwPmwxvsRpSmvThoK8

╭──────────────────────────────────────────────────────────────────────────────────────╮

│ Transaction Data │

├──────────────────────────────────────────────────────────────────────────────────────┤

│ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Owner: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Budget: 100000000 MIST │ │ Gas Price: 1000 MIST │ │ Gas Payment: │ │ ┌── │ │ │ ID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Version: 22 │ │ │ Digest: DiBrBMshDiD9cThpaEgpcYSF76uV4hCoE1qRyQ3rnYCB │ │ └── │ │ │ │ Transaction Kind: Programmable │ │

╭──────────────────────────────────────────────────────────────────────────────╮ │ │

│ Input Objects │ │ │

├──────────────────────────────────────────────────────────────────────────────┤ │ │

│ 0 Pure Arg: Type: address, Value: "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" │ │ │

╰──────────────────────────────────────────────────────────────────────────────╯ │ │

╭──────────────────────────────────────────────────────────────╮ │ │ │ Commands │ │ │
├──────────────────────────────────────────────────────────────┤ │ │ │ 0 MoveCall: │ │ │ │ ┌ │ │ │ │ │
Function: new │ │ │ │ │ Module: todo_list │ │ │ │ │ Package: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ │ └ │ │ │ │ │ │ │ │ │ 1 TransferObjects: │ │ │ │ ┌ │ │ │ │ │ Arguments: │ │ │ │ │ Result 0 │ │ │ │ │ Address: Input 0 │ │ │ │ └ │ │ │
╰──────────────────────────────────────────────────────────────╯ │ │ │ │ Signatures: │ │

C5Lie4dtP5d3OkKzFBa+xM0BiNoB/A4ItthDCRTRBUrEE+jXeNs7mP4AuGwi3nzfTskh29+R1j1Kba4Wdy3QDA== │ │ │
╰──────────────────────────────────────────────────────────────────────────────────────╯

╭──────────────────────────────────────────────────────────────────────────╮ │ Transaction Effects │ ├──────────────────────────────────────────────────────────────────────────┤ │
Digest: BJwYEnuuMzU4Y8cTwMoJbbQA6cLwPmwxvsRpSmvThoK8 │ │ Status: Success │ │ Executed Epoch: 1213 │ │ │ │ Created Objects: │ │ ┌── │ │ │ ID: 0x74973c4ea2e78dc409f60481e23761cee68a48156df93a93fbcceb77d1cacdf6 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 23 │ │ │ Digest: DuHTozDHMsuA7cFnWRQ1Gb8FQghAEBaj3inasJxqYq1c │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │ │ Gas Object: │ │ ┌── │ │ │ ID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │ │ Gas Cost Summary: │ │ Storage Cost: 2318000 MIST │ │ Computation Cost: 1000000 MIST │ │ Storage Rebate: 978120 MIST │ │ Non-refundable Storage Fee: 9880 MIST │ │ │ │ Transaction Dependencies: │ │ FSz2fYXmKqTf77mFXNq5JK7cKY8agWja7V5yDKEgL8c3 │ │ GgMZKTt482DYApbAZkPDtdssGHZLbxgjm2uMXhzJax8Q │
╰──────────────────────────────────────────────────────────────────────────╯

╭─────────────────────────────╮ │ No transaction block events │ ╰─────────────────────────────╯

╭──────────────────────────────────────────────────────────────────────────────╮ │ Object Changes │
├──────────────────────────────────────────────────────────────────────────────┤ │ Created Objects: │ │ ┌── │ │ │ ObjectID: 0x74973c4ea2e78dc409f60481e23761cee68a48156df93a93fbcceb77d1cacdf6 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ Version: 23 │ │ │ Digest: DuHTozDHMsuA7cFnWRQ1Gb8FQghAEBaj3inasJxqYq1c │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │
╰──────────────────────────────────────────────────────────────────────────────╯

╭──────────────────────────────────────────────────────────────────────────╮ │ Balance Changes │ │ ├──────────────────────────────────────────────────────────────────────────┤ │
┌── │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ CoinType: 0x2::sui::SUI │ │ │ Amount: -2339880 │ │ └── │
╰──────────────────────────────────────────────────────────────────────────╯ ```

The section that we want to focus on is the "Object Changes". More specifically, the "Created Objects" part of it. It contains the object ID, the type and the version of the TodoList that you have created. We will use this object ID to interact with the list.

```
bash ╭──────────────────────────────────────────────────────────────────────────────╮ │ Object Changes │
├──────────────────────────────────────────────────────────────────────────────┤ │ Created Objects: │ │ ┌── │ │ │ ObjectID:
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType:
0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ Version: 22 │ │ │ Digest:
HyWdUpjuhjLY38dLpg6KPHQ3bt4BqQAbdF5gB8HQdEqG │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │
Version: 22 │ │ │ Digest: DiBrBMshDiD9cThpaEgpcYSF76uV4hCoE1qRyQ3rnYCB │ │ └── │
╰──────────────────────────────────────────────────────────────────────────────╯
```

In this example the object ID is 0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 . And the owner should be your account address. We achieved this by transferring the object to the sender in the last command of the transaction.

Another way to test that you have successfully created the list is to check the account objects.

```
bash $ sui client objects
```

It should have an object that looks similar to this:

```
bash ╭ ... ╮ │ ╭─────────────┬──────────────────────────────────────────────────────────────╮ │ │ │ objectId │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ │ version │ 22 │ │ │ │ digest │ /DUEiCLkaNSgzpZSq2vSV0auQQEQhyH9occq9grMBZM= │ │
│ │ objectType │ 0x468d..29fe::todo_list::TodoList │ │ │ ╰─────────────┴──────────────────────────────────────────────────────────────╯ │ │ ... │
```

The TodoList that we created in the previous step is an object that you can interact with as its owner. You can call functions defined in the todo_list module on this object. To demonstrate this, we will add an item to the list. First, we will add just one item, and in the second transaction we will add 3 and remove another one.

Double check that you have variables set up [from the previous step](), and then add one more variable for the list object.

```
bash $ export LIST_ID=0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553
```

Now we can construct the transaction to add an item to the list. The command will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Finish the Hello, Sui chapter'"
```

In this command, we are calling the add function in the todo_list package. The function takes two arguments: the list object and the item to add. The item is a string, so we need to wrap it in single quotes. The command will add the item to the list.

If everything is correct, you should see the output similar to the one we had in previous sections. Now you can check the list object to see if the item was added.

```
bash $ sui client object $LIST_ID
```

The output should contain the item that you have added.

```
bash ╭──────────────────┬──────────────────────────────────────────────────────────────────────────────────────╮ │ objectId │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ version │ 24 │ │ digest │ FGcXH8MGpMs5BdTnC62CQ3VLAwwexYg2id5DKU7Jr9aQ │ │ objType
│ 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ owner │
╭──────────┬──────────────────────────────────────────────────────────────╮ │ │ │ │ AddressOwner │
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ │
╰──────────┴──────────────────────────────────────────────────────────────╯ │ │ prevTx │ EJVK6FEHtfTdCuGkNsU1HcrmUBEN6H6jshfcptnw8Yt1 │ │
storageRebate │ 1558000 │ │ content │ ╭──────────────────────┬──────────────────────────────────────────────────────────────────────╮
│ │ │ │ dataType │ moveObject │ │ │ │ │ type │ 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ │ │
hasPublicTransfer │ true │ │ │ │ │ fields │ ╭────────┬──────────────────────────────────────────────────────────────╮ │ │ │ │ │ │ │ id │
╭───┬──────────────────────────────────────────────────────────────────╮ │ │ │ │ │ │ │ │ │ │ id │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ │ │ │ │ │ │ │
╰───┴──────────────────────────────────────────────────────────────────╯ │ │ │ │ │ │ │ │ items │ ╭────────────────────────────╮ │ │ │ │ │ │ │
│ │ │ finish the Hello, Sui chapter │ │ │ │ │ │ │ │ │ │ ╰────────────────────────────╯ │ │ │ │ │ │ │
╰─────┴──────────────────────────────────────────────────────────────────────────────╯ │ │ │ │
╰───────────────────┴──────────────────────────────────────────────────────────────────────────────╯ │
╰──────────────┴──────────────────────────────────────────────────────────────────────────────────────────╯
```

A JSON representation of the object can be obtained by adding the --json flag to the command.

```
bash $ sui client object $LIST_ID --json
```

```
bash { "objectId": "0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553", "version": "24", "digest":
"FGcXH8MGpMs5BdTnC62CQ3VLAwwexYg2id5DKU7Jr9aQ", "type": "0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList",
"owner": { "AddressOwner": "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" }, "previousTransaction":
"EJVK6FEHtfTdCuGkNsU1HcrmUBEN6H6jshfcptnw8Yt1", "storageRebate": "1558000", "content": { "dataType": "moveObject", "type":
"0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList", "hasPublicTransfer": true, "fields": { "id": { "id":
"0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553" }, "items": ["Finish the Hello, Sui chapter"] } } }
```

You can chain multiple commands in a single transaction. This shows the power of Transaction Blocks! Using the same list object, we will add three more items and remove one. The command will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Finish Concepts chapter'" \ --move-call
$PACKAGE_ID::todo_list::add @$LIST_ID "'Read the Move Basics chapter'" \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Learn about Object Model'" \
--move-call $PACKAGE_ID::todo_list::remove @$LIST_ID 0
```

If previous commands were successful, this one should not be any different. You can check the list object to see if the items were added and removed. The JSON representation is a bit more readable!

```
bash sui client object $LIST_ID --json
```

```
bash { "objectId": "0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553", "version": "25", "digest":
"EDTXDsteqPGAGu4zFAj5bbQGTkucWk4hhuUquk39enGA", "type": "0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList",
"owner": { "AddressOwner": "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" }, "previousTransaction":
"7SXLGBSh31jv8G7okQ9mEgnw5MnTfvzzHEHpWf3Sa9gY", "storageRebate": "1922800", "content": { "dataType": "moveObject", "type":
"0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList", "hasPublicTransfer": true, "fields": { "id": { "id":
"0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553" }, "items": [ "Finish Concepts chapter", "Read the Move Basics chapter", "Learn
about Object Model" ] } } }
```

Commands don't have to be in the same package or operate on the same object. Within a single transaction block, you can interact with multiple packages and objects. This is a powerful feature that allows you to build complex interactions on-chain!

In this guide, we have shown how to publish a package on the Move blockchain and interact with it using the Sui CLI. We have demonstrated how to create a new list object, add items to it, and remove them. We have also shown how to chain multiple commands in a single transaction block. This guide should give you a good starting point for building your own applications on the Sui blockchain!

## Requesting Coins

In devnet and testnet environments, the CLI provides a way to request coins to your account, so you can interact with the network. To request coins, run the sui client faucet command:

```
bash $ sui client faucet Request successful. It can take up to 1 minute to get the coin. Run sui client gas to check your gas coins.
```

After waiting a little bit, you can check that the Coin object was sent to your account by running the sui client balance command:

```
bash $ sui client balance ╭──────────────────────────────────────╮ │ Balance of coins owned by this address │
├──────────────────────────────────────┤ │ ╭──────────────────────────────╮ │ │ │ coin balance (raw) balance │ │ │
├──────────────────────────────────┤ │ │ │ Sui 1000000000 1.00 SUI │ │ │ ╰──────────────────────────────╯ │
╰──────────────────────────────────────╯
```

Alternatively, you can query objects owned by your account, by running the sui client objects command. The actual output will be different, because the object ID is unique, and so is digest, but the structure will be similar:

```
bash $ sui client objects ╭──────────────────────────────────────────────────────────────────────────────────╮ │
╭──────────────┬──────────────────────────────────────────────────────────────────╮ │ │ │ objectId │
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ │ version │ 4 │ │ │ │ digest │ nA68oa8gab/CdIRw+240wze8u0P+sRe4vcisbENcR4U= │ │
│ │ objectType │ 0x0000..0002::coin::Coin │ │ │ ╰──────────────┴──────────────────────────────────────────────────╯ │
╰──────────────────────────────────────────────────────────────────────────────────╯
```

Now that we have the account set up and the coins in the account, we can interact with the network. We will start by publishing the package to the network.

To publish the package to the network, we will use the sui client publish command. The command will automatically build the package and use its bytecode to publish in a single transaction.

We are using the --gas-budget argument during publishing. It specifies how much gas we are willing to spend on the transaction. We won't touch on this topic in this section, but it's important to know that every transaction in Sui costs gas, and the gas is paid in SUI coins.

The gas-budget is specified in MISTs . 1 SUI equals 10^9 MISTs. For the sake of demonstration, we will use 100,000,000 MISTs, which is 0.1 SUI.

```bash

# run this from the `todo_list` folder
```

$ sui client publish --gas-budget 100000000

## alternatively, you can specify path to the package

$ sui client publish --gas-budget 100000000 todo_list ```

The output of the publish command is rather lengthy, so we will show and explain it in parts.

```
bash $ sui client publish --gas-budget 100000000 UPDATING GIT DEPENDENCY https://github.com/MystenLabs/sui.git INCLUDING DEPENDENCY Bridge INCLUDING
DEPENDENCY DeepBook INCLUDING DEPENDENCY SuiSystem INCLUDING DEPENDENCY Sui INCLUDING DEPENDENCY MoveStdlib BUILDING todo_list Successfully verified
dependencies on-chain against source. Transaction Digest: GpcDV6JjjGQMRwHpEz582qsd5MpCYgSwrDAq1JXcpFjW
```

As you can see, when we run the publish command, the CLI first builds the package, then verifies the dependencies on-chain, and finally publishes the package. The output of the command is the transaction digest, which is a unique identifier of the transaction and can be used to query the transaction status.

The section titled TransactionData contains the information about the transaction we just sent. It features fields like sender , which is your address, the gas_budget set with the --gas-budget argument, and the Coin we used for payment. It also prints the Commands that were run by the CLI. In this example, the commands Publish and TransferObject were run - the latter transfers a special object UpgradeCap to the sender.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────────────────────╮ │ Transaction Data │
├──────────────────────────────────────────────────────────────────────────────────────────────────────────────┤ │ Sender:
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Owner: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │
Gas Budget: 100000000 MIST │ │ Gas Price: 1000 MIST │ │ Gas Payment: │ │ ┌── │ │ │ ID: 0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de
│ │ │ Version: 7 │ │ │ Digest: AXYPnups8A5J6pkvLa6RekX2ye3qur66EZ88mEbaUDQ1 │ │ └── │ │ │ │ Transaction Kind: Programmable │ │
╭────────────────────────────────────────────────────────────────────────────────────────────────────────────╮ │ │ │ Input Objects │ │ │
├────────────────────────────────────────────────────────────────────────────────────────────────────────────┤ │ │ │ 0 Pure Arg: Type: address, Value:
"0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" │ │ │
╰────────────────────────────────────────────────────────────────────────────────────────────────────────────╯ │ │
╭────────────────────────────────────────────────────────────────────────────╮ │ │ │ Commands │ │ │
├────────────────────────────────────────────────────────────────────────────┤ │ │ │ 0 Publish: │ │ │ │ ┌ │ │ │ │ │ Dependencies: │ │ │ │ │
0x0000000000000000000000000000000000000000000000000000000000000001 │ │ │ │ │ 0x0000000000000000000000000000000000000000000000000000000000000002 │ │ │ │ └
│ │ │ │ │ │ │ │ 1 TransferObjects: │ │ │ │ ┌ │ │ │ │ │ Arguments: │ │ │ │ │ Result 0 │ │ │ │ │ Address: Input 0 │ │ │ │ └ │ │ │
╰────────────────────────────────────────────────────────────────────────────╯ │ │ │ │ Signatures: │ │
gebjSbVwZwTkizfYg2XIuzdx+d66VxFz8EmVaisVFiV3GkDay6L+hQG3n2CQ1hrWphP6ZLc7bd1WRq4ss+hQAQ== │ │ │
╰──────────────────────────────────────────────────────────────────────────────────────────────────────────────╯
```

Transaction Effects contains the status of the transaction, the changes that the transaction made to the state of the network and the objects involved in the transaction.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────────────────╮ │ Transaction Effects │
├──────────────────────────────────────────────────────────────────────────────────────────────────────┤ │ Digest:
GpcDV6JjjGQMRwHpEz582qsd5MpCYgSwrDAq1JXcpFjW │ │ Status: Success │ │ Executed Epoch: 411 │ │ │ │ Created Objects: │ │ ┌── │ │ │ ID:
0x160f7856e13b27e5a025112f361370f4efc2c2659cb0023f1e99a8a84d1652f3 │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 8y6bhwvQrGJHDckUZmj2HDAjfkyVqHohhvY1Fvzyj7ec │ │ └──
│ │ ┌── │ │ │ ID: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ Owner: Immutable │ │ │ Version: 1 │ │ │ Digest:
Ein91NF2hc3qC4XYoMUFMfin9U23xQmDAdEMSHLae7MK │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ID:
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └──
│ │ Gas Object: │ │ ┌── │ │ │ ID: 0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └──
│ │ Gas Cost Summary: │ │ Storage Cost: 10404400 MIST │ │ Computation Cost: 1000000 MIST │ │ Storage Rebate: 978120 MIST │ │ Non-refundable Storage Fee:
9880 MIST │ │ │ │ Transaction Dependencies: │ │ 7Ukrc5GqdFqTA41wvWgreCdHn2vRLfgQ3YMFkdks72Vk │ │ 7d4amuHGhjtYKujEs9YkJARzNEn4mRbWWv3fn4cdKdyh │
╰──────────────────────────────────────────────────────────────────────────────────────────────────────╯
```

If there were any events emitted, you would see them in this section. Our package does not use events, so the section is empty.

```
bash ╭─────────────────────────────╮ │ No transaction block events │ ╰─────────────────────────────╯
```

These are the changes to objects that transaction has made. In our case, we have created a new UpgradeCap object which is a special object that allows the sender to upgrade the package in the future, mutated the Gas object, and published a new package. Packages are also objects on Sui.

```
bash ╭─────────────────────────────────────────────────────────────────────────────────────────────────╮ │ Object Changes │
├──────────────────────────────────────────────────────────────────────────────────────────────────────┤ │ Created Objects: │ │ ┌── │ │ │ ObjectID:
0x160f7856e13b27e5a025112f361370f4efc2c2659cb0023f1e99a8a84d1652f3 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::package::UpgradeCap │ │ │ Version: 8
│ │ │ Digest: 8y6bhwvQrGJHDckUZmj2HDAjfkyVqHohhvY1Fvzyj7ec │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │
Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └── │ │ Published Objects: │ │ ┌── │ │ │ PackageID:
0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ Version: 1 │ │ │ Digest: Ein91NF2hc3qC4XYoMUFMfin9U23xQmDAdEMSHLae7MK │ │ │
Modules: todo_list │ │ └── │ ╰─────────────────────────────────────────────────────────────────────────────────╯
```

This last section contains changes to SUI Coins, in our case, we have spent around 0.015 SUI, which in MIST is 10,500,000. You can see it under the amount field in the output.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────────────────╮ │ Balance Changes │
├──────────────────────────────────────────────────────────────────────────────────────────────────────┤ │ ┌── │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ CoinType: 0x2::sui::SUI │ │ │ Amount: -10426280 │ │ └── │
╰──────────────────────────────────────────────────────────────────────────────────────────────────────╯
```

It is possible to specify the --json flag during publishing to get the output in JSON format. This is useful if you want to parse the output programmatically or store it for later use.

```
bash $ sui client publish --gas-budget 100000000 --json
```

After the package is published on chain, we can interact with it. To do this, we need to find the address (object ID) of the package. It's under the Published Objects section of the Object Changes output. The address is unique for each package, so you will need to copy it from the output.

In this example, the address is:

```
bash 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe
```

Now that we have the address, we can interact with the package. In the next section, we will show how to interact with the package by sending transactions.

To demonstrate the interaction with the todo_list package, we will send a transaction to create a new list and add an item to it. Transactions are sent via the sui client ptb command, it allows using the Transaction Blocks at full capacity. The command may look big and complex, but we go through it step by step.

Before we construct the command, let's store the values we will use in the transaction. Replace the 0x4.... with the address of the package you have published. And MY_ADDRESS variable will be automatically set to your address from the CLI output.

```
bash $ export PACKAGE_ID=0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe $ export MY_ADDRESS=$(sui client active-address)
```

Now to building an actual transaction. The transaction will consist of two parts: we will call the new function in the todo_list package to create a new list, and then we will transfer the list object to our account. The transaction will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --assign sender @$MY_ADDRESS \ --move-call $PACKAGE_ID::todo_list::new \ --assign list \ --transfer-
objects "[list]" sender
```

In this command, we are using the ptb subcommand to build a transaction. Parameters that follow it define the actual commands and actions that the transaction will perform. The first two calls we make are utility calls to set the sender address to the command inputs and set the gas budget for the transaction.

~~~bash

## sets the gas budget for the transaction

--gas-budget 100000000 \n

## registers a variable "sender=@..."

--assign sender @$MY_ADDRESS \n ```

Then we perform the actual call to a function in the package. We use the --move-call followed by the package ID, the module name, and the function name. In this case, we are calling the new function in the todo_list package.

```bash

## calls the "new" function in the "todo_list" package under the $PACKAGE_ID address

--move-call $PACKAGE_ID::todo_list::new ```

The function that we defined actually returns a value, which we want need to store. We use the --assign command to give a name to the returned value. In this case, we are calling it list . And then we transfer the object to our account.

```bash --move-call $PACKAGE_ID::todo_list::new \

## assigns the result of the "new" function to the "list" variable (from the previous step)

--assign list \

## transfers the object to the sender

--transfer-objects "[list]" sender ```

Once the command is constructed, you can run it in the terminal. If everything is correct, you should see the output similar to the one we had in previous sections. The output will contain the transaction digest, the transaction data, and the transaction effects.

```bash Transaction Digest: BJwYEnuuMzU4Y8cTwMoJbbQA6cLwPmwxvsRpSmvThoK8

╭──────────────────────────────────────────────────────────────────────╮

│ Transaction Data │

├──────────────────────────────────────────────────────────────────────┤

│ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Owner: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Budget: 100000000 MIST │ │ Gas Price: 1000 MIST │ │ Gas Payment: │ │ ┌── │ │ │ ID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Version: 22 │ │ │ Digest: DiBrBMshDiD9cThpaEgpcYSF76uV4hCoE1qRyQ3rnYCB │ │ └── │ │ │ │ Transaction Kind: Programmable │ │

╭──────────────────────────────────────────────────────────────────╮ │ │

│ Input Objects │ │ │

├──────────────────────────────────────────────────────────────────┤ │ │

│ 0 Pure Arg: Type: address, Value: "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" │ │ │

╰──────────────────────────────────────────────────────────────────╯ │ │

╭──────────────────────────────────────────────────────────╮ │ │ │ Commands │ │ │
├──────────────────────────────────────────────────────────┤ │ │ │ 0 MoveCall: │ │ │ │ ┌ │ │ │ │ │
Function: new │ │ │ │ │ Module: todo_list │ │ │ │ │ Package: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ │ └ │ │ │ │ │ │ │ │ │ 1 TransferObjects: │ │ │ │ ┌ │ │ │ │ │ │ Arguments: │ │ │ │ │ Result 0 │ │ │ │ │ │ Address: Input 0 │ │ │ │ └ │ │ │
╰──────────────────────────────────────────────────────────╯ │ │ │ │ Signatures: │ │

C5Lie4dtP5d3OkKzFBa+xM0BiNoB/A4IthDCRTRBUrEE+jXeNs7mP4AuGwi3nzfTskh29+R1j1Kba4Wdy3QDA== │ │ │

╰──────────────────────────────────────────────────────────────────────╯

╭──────────────────────────────────────────────────────────────────╮ │ Transaction Effects │ ├──────────────────────────────────────────────────────────────────┤ │
Digest: BJwYEnuuMzU4Y8cTwMoJbbQA6cLwPmwxvsRpSmvThoK8 │ │ Status: Success │ │ Executed Epoch: 1213 │ │ │ │ Created Objects: │ │ ┌── │ │ │ ID: 0x74973c4ea2e78dc409f60481e23761cee68a48156df93a93fbcceb77d1cacdf6 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 23 │ │ │ Digest: DuHTozDHMsuA7cFnWRQ1Gb8FQghAEBaj3inasJxqYq1c │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │ │ Gas Object: │ │ ┌── │ │ │ ID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │ │ Gas Cost Summary: │ │ Storage Cost: 2318000 MIST │ │ Computation Cost: 1000000 MIST │ │ Storage Rebate: 978120 MIST │ │ Non-refundable Storage Fee: 9880 MIST │ │ │ │ Transaction Dependencies: │ │ FSz2fYXmKqTf77mFXNq5JK7cKY8agWja7V5yDKEgL8c3 │ │ GgMZKTt482DYApbAZkPDtdssGHZLbxgjm2uMXhzJax8Q │
╰──────────────────────────────────────────────────────────────────╯

╭──────────────────────────╮ │ No transaction block events │ ╰──────────────────────────╯

╭──────────────────────────────────────────────────────────────────────╮ │ Object Changes │
├──────────────────────────────────────────────────────────────────────┤ │ Created Objects: │ │ ┌── │ │ │ ObjectID: 0x74973c4ea2e78dc409f60481e23761cee68a48156df93a93fbcceb77d1cacdf6 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ Version: 23 │ │ │ Digest: DuHTozDHMsuA7cFnWRQ1Gb8FQghAEBaj3inasJxqYq1c │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │
╰──────────────────────────────────────────────────────────────────────╯

╭──────────────────────────────────────────────────────────────────╮ │ Balance Changes │ ├──────────────────────────────────────────────────────────────────┤ │ ┌── │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ CoinType: 0x2::sui::SUI │ │ │ Amount: -2339880 │ │ └── │
╰──────────────────────────────────────────────────────────────────╯ ```

The section that we want to focus on is the "Object Changes". More specifically, the "Created Objects" part of it. It contains the object ID, the type and the version of the TodoList that you have created. We will use this object ID to interact with the list.

~~~
bash ╭──────────────────────────────────────────────────────────────────╮ │ Object Changes │
├──────────────────────────────────────────────────────────────────┤ │ Created Objects: │ │ ┌── │ │ │ ObjectID:
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType:
```
```

```
0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ Version: 22 │ │ │ Digest:
HyWdUpjuhjLY38dLpg6KPHQ3bt4BqQAbdF5gB8HQdEqG │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │
Version: 22 │ │ │ Digest: DiBrBMshDiD9cThpaEgpcYSF76uV4hCoE1qRyQ3rnYCB │ │ └── │
╰──────────────────────────────────────────────────────────────────────────────────────────────╯
```

In this example the object ID is 0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 . And the owner should be your account address. We achieved this by transferring the object to the sender in the last command of the transaction.

Another way to test that you have successfully created the list is to check the account objects.

```
bash $ sui client objects
```

It should have an object that looks similar to this:

```
bash ╭ ... ╮ │ ╭──────────────┬──────────────────────────────────────────────────────╮ │ │ │ objectId │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ │ version │ 22 │ │ │ │ digest │ /DUEiCLkaNSgzpZSq2vSV0auQQEQhyH9occq9grMBZM= │ │
│ │ objectType │ 0x468d..29fe::todo_list::TodoList │ │ │ ╰──────────────┴──────────────────────────────────────────────────────╯ │ │ ... │
```

The TodoList that we created in the previous step is an object that you can interact with as its owner. You can call functions defined in the todo_list module on this object. To demonstrate this, we will add an item to the list. First, we will add just one item, and in the second transaction we will add 3 and remove another one.

Double check that you have variables set up from the previous step , and then add one more variable for the list object.

```
bash $ export LIST_ID=0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553
```

Now we can construct the transaction to add an item to the list. The command will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Finish the Hello, Sui chapter'"
```

In this command, we are calling the add function in the todo_list package. The function takes two arguments: the list object and the item to add. The item is a string, so we need to wrap it in single quotes. The command will add the item to the list.

If everything is correct, you should see the output similar to the one we had in previous sections. Now you can check the list object to see if the item was added.

```
bash $ sui client object $LIST_ID
```

The output should contain the item that you have added.

```
bash ╭───────────────┬──────────────────────────────────────────────────────────────────────────────────────────╮ │ objectId │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ version │ 24 │ │ digest │ FGcXH8MGpMs5BdTnC62CQ3VLAwwexYg2id5DKU7Jr9aQ │ │ objType
│ 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ owner │
╭───────────┬──────────────────────────────────────────────────────────────────────╮ │ │ │ │ AddressOwner │
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ │
╰───────────┴──────────────────────────────────────────────────────────────────────╯ │ │ prevTx │ EJVK6FEHtfTdCuGkNsU1HcrmUBEN6H6jshfcptnw8Yt1 │ │
storageRebate │ 1558000 │ │ content │ ╭───────────────────┬──────────────────────────────────────────────────────────────╮
│ │ │ │ dataType │ moveObject │ │ │ │ │ type │ 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ │ │
hasPublicTransfer │ true │ │ │ │ │ fields │ ╭────────┬────────────────────────────────────────────────────────╮ │ │ │ │ │ │ │ id │
╭─────┬───────────────────────────────────────────────────────────────────╮ │ │ │ │ │ │ │ │ │ │ id │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ │ │ │ │ │ │ │
╰─────┴───────────────────────────────────────────────────────────────────╯ │ │ │ │ │ │ │ │ items │ ╭──────────────────────────────────╮ │ │ │ │ │ │ │
│ │ │ finish the Hello, Sui chapter │ │ │ │ │ │ │ │ │ │ ╰──────────────────────────────────╯ │ │ │ │ │ │ │ │
╰────────┴───────────────────────────────────────────────────────────────────────────────────╯ │ │ │ │
╰───────────────────┴──────────────────────────────────────────────────────────────────────────────────────────────╯ │
╰───────────────┴───────────────────────────────────────────────────────────────────────────────────────────────────────────╯
```

A JSON representation of the object can be obtained by adding the --json flag to the command.

```
bash $ sui client object $LIST_ID --json
```

```
bash { "objectId": "0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553", "version": "24", "digest":
"FGcXH8MGpMs5BdTnC62CQ3VLAwwexYg2id5DKU7Jr9aQ", "type": "0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList",
"owner": { "AddressOwner": "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" }, "previousTransaction":
"EJVK6FEHtfTdCuGkNsU1HcrmUBEN6H6jshfcptnw8Yt1", "storageRebate": "1558000", "content": { "dataType": "moveObject", "type":
"0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList", "hasPublicTransfer": true, "fields": { "id": { "id":
"0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553" }, "items": ["Finish the Hello, Sui chapter"] } } }
```

You can chain multiple commands in a single transaction. This shows the power of Transaction Blocks! Using the same list object, we will add three more items and remove one. The command will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Finish Concepts chapter'" \ --move-call
$PACKAGE_ID::todo_list::add @$LIST_ID "'Read the Move Basics chapter'" \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Learn about Object Model'" \
--move-call $PACKAGE_ID::todo_list::remove @$LIST_ID 0
```

If previous commands were successful, this one should not be any different. You can check the list object to see if the items were added and removed. The JSON representation is a bit more readable!

```
bash sui client object $LIST_ID --json
```

```
bash { "objectId": "0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553", "version": "25", "digest":
"EDTXDsteqPGAGu4zFAj5bbQGTkucWk4hhuUquk39enGA", "type": "0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList",
"owner": { "AddressOwner": "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" }, "previousTransaction":
"7SXLGBSh31jv8G7okQ9mEgnw5MnTfvzzHEHpWf3Sa9gY", "storageRebate": "1922800", "content": { "dataType": "moveObject", "type":
"0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList", "hasPublicTransfer": true, "fields": { "id": { "id":
"0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553" }, "items": [ "Finish Concepts chapter", "Read the Move Basics chapter", "Learn
about Object Model" ] } } }
```

Commands don't have to be in the same package or operate on the same object. Within a single transaction block, you can interact with multiple packages and objects. This is a powerful feature that allows you to build complex interactions on-chain!

In this guide, we have shown how to publish a package on the Move blockchain and interact with it using the Sui CLI. We have demonstrated how to create a new list object, add items to it, and remove them. We have also shown how to chain multiple commands in a single transaction block. This guide should give you a good starting point for building your own applications on the Sui blockchain!

## Publish

To publish the package to the network, we will use the sui client publish command. The command will automatically build the package and use its bytecode to publish in a single transaction.

We are using the --gas-budget argument during publishing. It specifies how much gas we are willing to spend on the transaction. We won't touch on this topic in this section, but it's important to know that every transaction in Sui costs gas, and the gas is paid in SUI coins.

The gas-budget is specified in MISTs . 1 SUI equals 10^9 MISTs. For the sake of demonstration, we will use 100,000,000 MISTs, which is 0.1 SUI.

```bash

# run this from the `todo_list` folder

$ sui client publish --gas-budget 100000000
```

## alternatively, you can specify path to the package

$ sui client publish --gas-budget 100000000 todo_list ```

The output of the publish command is rather lengthy, so we will show and explain it in parts.

```
bash $ sui client publish --gas-budget 100000000 UPDATING GIT DEPENDENCY https://github.com/MystenLabs/sui.git INCLUDING DEPENDENCY Bridge INCLUDING
DEPENDENCY DeepBook INCLUDING DEPENDENCY SuiSystem INCLUDING DEPENDENCY Sui INCLUDING DEPENDENCY MoveStdlib BUILDING todo_list Successfully verified
dependencies on-chain against source. Transaction Digest: GpcDV6JjjGQMRwHpEz582qsd5MpCYgSwrDAq1JXcpFjW
```

As you can see, when we run the publish command, the CLI first builds the package, then verifies the dependencies on-chain, and finally publishes the package. The output of the command is the transaction digest, which is a unique identifier of the transaction and can be used to query the transaction status.

The section titled TransactionData contains the information about the transaction we just sent. It features fields like sender , which is your address, the gas_budget set with the --gas-budget argument, and the Coin we used for payment. It also prints the Commands that were run by the CLI. In this example, the commands Publish and TransferObject were run - the latter transfers a special object UpgradeCap to the sender.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────────────╮ │ Transaction Data │
├──────────────────────────────────────────────────────────────────────────────────────────────┤ │ Sender:
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Owner: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │
Gas Budget: 100000000 MIST │ │ Gas Price: 1000 MIST │ │ Gas Payment: │ │ ┌── │ │ │ ID: 0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de
│ │ │ Version: 7 │ │ │ Digest: AXYPnups8A5J6pkvLa6RekX2ye3qur66EZ88mEbaUDQ1 │ │ └── │ │ │ │ Transaction Kind: Programmable │ │
╭──────────────────────────────────────────────────────────────────────────────────────────╮ │ │ │ Input Objects │ │ │
├──────────────────────────────────────────────────────────────────────────────────────────┤ │ │ │ 0 Pure Arg: Type: address, Value:
"0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" │ │ │
╰──────────────────────────────────────────────────────────────────────────────────────────╯ │ │
╭──────────────────────────────────────────────────────────────────────╮ │ │ │ Commands │ │ │
├──────────────────────────────────────────────────────────────────────┤ │ │ │ 0 Publish: │ │ │ │ ┌ │ │ │ │ │ Dependencies: │ │ │ │ │
0x0000000000000000000000000000000000000000000000000000000000000001 │ │ │ │ │ 0x0000000000000000000000000000000000000000000000000000000000000002 │ │ │ │ └
│ │ │ │ │ │ │ │ 1 TransferObjects: │ │ │ │ ┌ │ │ │ │ │ Arguments: │ │ │ │ │ Result 0 │ │ │ │ │ Address: Input 0 │ │ │ │ └ │ │ │
╰──────────────────────────────────────────────────────────────────────╯ │ │ │ │ Signatures: │ │
gebjSbVwZwTkizfYg2XIuzdx+d66VxFz8EmVaisVFiV3GkDay6L+hQG3n2CQ1hrWphP6ZLc7bd1WRq4ss+hQAQ== │ │ │
╰──────────────────────────────────────────────────────────────────────────────────────────────╯
```

Transaction Effects contains the status of the transaction, the changes that the transaction made to the state of the network and the objects involved in the transaction.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────╮ │ Transaction Effects │
├──────────────────────────────────────────────────────────────────────────────────────┤ │ Digest:
GpcDV6JjjGQMRwHpEz582qsd5MpCYgSwrDAq1JXcpFjW │ │ Status: Success │ │ Executed Epoch: 411 │ │ │ │ Created Objects: │ │ ┌── │ │ │ ID:
0x160f7856e13b27e5a025112f361370f4efc2c2659cb0023f1e99a8a84d1652f3 │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 8y6bhwvQrGJHDckUZmj2HDAjfkyVqHohhvY1Fvzyj7ec │ │ └──
│ │ ┌── │ │ │ ID: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ Owner: Immutable │ │ │ Version: 1 │ │ │ Digest:
Ein91NF2hc3qC4XYoMUFMfin9U23xQmDAdEMSHLae7MK │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ID:
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └──
│ │ Gas Object: │ │ ┌── │ │ │ ID: 0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └──
│ │ Gas Cost Summary: │ │ Storage Cost: 10404400 MIST │ │ Computation Cost: 1000000 MIST │ │ Storage Rebate: 978120 MIST │ │ Non-refundable Storage Fee:
9880 MIST │ │ │ │ Transaction Dependencies: │ │ 7Ukrc5GqdFqTA41wvWgreCdHn2vRLfgQ3YMFkdks72Vk │ │ 7d4amuHGhjtYKujEs9YkJARzNEn4mRbWWv3fn4cdKdyh │
╰──────────────────────────────────────────────────────────────────────────────────────╯
```

If there were any events emitted, you would see them in this section. Our package does not use events, so the section is empty.

```
bash ╭─────────────────────────────╮ │ No transaction block events │ ╰─────────────────────────────╯
```

These are the changes to objects that transaction has made. In our case, we have created a new UpgradeCap object which is a special object that allows the sender to upgrade the package in the future, mutated the Gas object, and published a new package. Packages are also objects on Sui.

```
bash ╭─────────────────────────────────────────────────────────────────────────────────────╮ │ Object Changes │
├─────────────────────────────────────────────────────────────────────────────────────┤ │ Created Objects: │ │ ┌── │ │ │ ObjectID:
0x160f7856e13b27e5a025112f361370f4efc2c2659cb0023f1e99a8a84d1652f3 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::package::UpgradeCap │ │ │ Version: 8
│ │ │ Digest: 8y6bhwvQrGJHDckUZmj2HDAjfkyVqHohhvY1Fvzyj7ec │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0x4ea1303e4f5e2f65fc3709bc0fb70a3035fdd2d53dbcff33e026a50a742ce0de │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │
Version: 8 │ │ │ Digest: 7ydahjaM47Gyb33PB4qnW2ZAGqZvDuWScV6sWPiv7LTc │ │ └── │ │ Published Objects: │ │ ┌── │ │ │ PackageID:
0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ Version: 1 │ │ │ Digest: Ein91NF2hc3qC4XYoMUFMfin9U23xQmDAdEMSHLae7MK │ │ │
Modules: todo_list │ │ └── │ ╰─────────────────────────────────────────────────────────────────────────────────────╯
```

This last section contains changes to SUI Coins, in our case, we have spent around 0.015 SUI, which in MIST is 10,500,000. You can see it under the amount field in the output.

```
bash ╭──────────────────────────────────────────────────────────────────────────────────────╮ │ Balance Changes │
├──────────────────────────────────────────────────────────────────────────────────────┤ │ ┌── │ │ │ Owner: Account Address (
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ CoinType: 0x2::sui::SUI │ │ │ Amount: -10426280 │ │ └── │
╰──────────────────────────────────────────────────────────────────────────────────────╯
```

It is possible to specify the --json flag during publishing to get the output in JSON format. This is useful if you want to parse the output programmatically or store it for later use.

```
bash $ sui client publish --gas-budget 100000000 --json
```

After the package is published on chain, we can interact with it. To do this, we need to find the address (object ID) of the package. It's under the Published Objects section of the Object Changes output. The address is unique for each package, so you will need to copy it from the output.

In this example, the address is:

```
bash 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe
```

Now that we have the address, we can interact with the package. In the next section, we will show how to interact with the package by sending transactions.

To demonstrate the interaction with the todo_list package, we will send a transaction to create a new list and add an item to it. Transactions are sent via the sui client ptb command, it allows using the Transaction Blocks at full capacity. The command may look big and complex, but we go through it step by step.

Before we construct the command, let's store the values we will use in the transaction. Replace the 0x4.... with the address of the package you have published. And MY_ADDRESS variable will be automatically set to your address from the CLI output.

```
bash $ export PACKAGE_ID=0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe $ export MY_ADDRESS=$(sui client active-address)
```

Now to building an actual transaction. The transaction will consist of two parts: we will call the new function in the todo_list package to create a new list, and then we will transfer the list object to our account. The transaction will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --assign sender @$MY_ADDRESS \ --move-call $PACKAGE_ID::todo_list::new \ --assign list \ --transfer-
objects "[list]" sender
```

In this command, we are using the ptb subcommand to build a transaction. Parameters that follow it define the actual commands and actions that the transaction will perform. The first two calls we make are utility calls to set the sender address to the command inputs and set the gas budget for the transaction.

```bash
```

## sets the gas budget for the transaction

--gas-budget 100000000 \n

## registers a variable "sender=@..."

--assign sender @$MY_ADDRESS \n ```

Then we perform the actual call to a function in the package. We use the --move-call followed by the package ID, the module name, and the function name. In this case, we are calling the new function in the todo_list package.

~~~bash

## calls the "new" function in the "todo_list" package under the $PACKAGE_ID address

--move-call $PACKAGE_ID::todo_list::new ```

The function that we defined actually returns a value, which we want need to store. We use the --assign command to give a name to the returned value. In this case, we are calling it list . And then we transfer the object to our account.

```bash --move-call $PACKAGE_ID::todo_list::new \

## assigns the result of the "new" function to the "list" variable (from the previous step)

--assign list \

## transfers the object to the sender

--transfer-objects "[list]" sender ```

Once the command is constructed, you can run it in the terminal. If everything is correct, you should see the output similar to the one we had in previous sections. The output will contain the transaction digest, the transaction data, and the transaction effects.

```bash Transaction Digest: BJwYEnuuMzU4Y8cTwMoJbbQA6cLwPmwxvsRpSmvThoK8

╭──────────────────────────────────────────────────────────────────────────────────────╮

│ Transaction Data │

├──────────────────────────────────────────────────────────────────────────────────────┤

│ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Owner: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Budget: 100000000 MIST │ │ Gas Price: 1000 MIST │ │ Gas Payment: │ │ ┌── │ │ │ ID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Version: 22 │ │ │ Digest: DiBrBMshDiD9cThpaEgpcYSF76uV4hCoE1qRyQ3rnYCB │ │ └── │ │ │ │ Transaction Kind: Programmable │ │

╭──────────────────────────────────────────────────────────────────────────────────╮ │ │

│ Input Objects │ │ │

├──────────────────────────────────────────────────────────────────────────────────┤ │ │

│ 0 Pure Arg: Type: address, Value: "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" │ │ │

╰──────────────────────────────────────────────────────────────────────────────────╯ │ │

╭──────────────────────────────────────────────────────────────────╮ │ │ │ Commands │ │ │
├──────────────────────────────────────────────────────────────────┤ │ │ │ 0 MoveCall: │ │ │ │ ┌ │ │ │ │ │
Function: new │ │ │ │ │ Module: todo_list │ │ │ │ │ Package: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ │ └ │ │ │ │ │ │ │ │ │ 1 TransferObjects: │ │ │ │ ┌ │ │ │ │ │ Arguments: │ │ │ │ │ Result 0 │ │ │ │ │ Address: Input 0 │ │ │ │ └ │ │ │
╰──────────────────────────────────────────────────────────────────╯ │ │ │ │ Signatures: │ │

C5Lie4dtP5d3OkKzFBa+xM0BiNoB/A4ItthDCRTRBUrEE+jXeNs7mP4AuGwi3nzfTskh29+R1j1Kba4Wdy3QDA== │ │ │
╰──────────────────────────────────────────────────────────────────────────────────────╯

╭──────────────────────────────────────────────────────────────────────────────╮ │ Transaction Effects │ ├──────────────────────────────────────────────────────────────────────────────┤ │
Digest: BJwYEnuuMzU4Y8cTwMoJbbQA6cLwPmwxvsRpSmvThoK8 │ │ Status: Success │ │ Executed Epoch: 1213 │ │ │ │ Created Objects: │ │ ┌── │ │ │ ID: 0x74973c4ea2e78dc409f60481e23761cee68a48156df93a93fbcceb77d1cacdf6 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 23 │ │ │ Digest: DuHTozDHMsuA7cFnWRQ1Gb8FQghAEBaj3inasJxqYq1c │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │ │ Gas Object: │ │ ┌── │ │ │ ID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │ │ Gas Cost Summary: │ │ Storage Cost: 2318000 MIST │ │ Computation Cost: 1000000 MIST │ │ Storage Rebate: 978120 MIST │ │ Non-refundable Storage Fee: 9880 MIST │ │ │ │ Transaction Dependencies: │ │ FSz2fYXmKqTf77mFXNq5JK7cKY8agWja7V5yDKEgL8c3 │ │ GgMZKTt482DYApbAZkPDtdssGHZLbxgjm2uMXhzJax8Q │
╰──────────────────────────────────────────────────────────────────────────────╯

╭──────────────────────────────╮ │ No transaction block events │ ╰──────────────────────────────╯

╭────────────────────────────────────────────────────────────────────────────────╮ │ Object Changes │
├────────────────────────────────────────────────────────────────────────────────┤ │ Created Objects: │ │ ┌── │ │ │ ObjectID: 0x74973c4ea2e78dc409f60481e23761cee68a48156df93a93fbcceb77d1cacdf6 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ Version: 23 │ │ │ Digest: DuHTozDHMsuA7cFnWRQ1Gb8FQghAEBaj3inasJxqYq1c │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │
╰────────────────────────────────────────────────────────────────────────────────╯

╭──────────────────────────────────────────────────────────────────────────────╮ │ Balance Changes │ │ ├──────────────────────────────────────────────────────────────────────────────┤ │
┌── │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ CoinType: 0x2::sui::SUI │ │ │ Amount: -2339880 │ │ └── │
╰──────────────────────────────────────────────────────────────────────────────╯ ```

The section that we want to focus on is the "Object Changes". More specifically, the "Created Objects" part of it. It contains the object ID, the type and the version of the TodoList that you have created. We will use this object ID to interact with the list.

~~~
bash ╭──────────────────────────────────────────────────────────────────────────────────────╮ │ Object Changes │
├──────────────────────────────────────────────────────────────────────────────────────┤ │ Created Objects: │ │ ┌── │ │ │ ObjectID:
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType:
0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ Version: 22 │ │ │ Digest:
HyWdUpjuhjLY38dLpg6KPHQ3bt4BqQAbdF5gB8HQdEqG │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
```
```

```
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │
Version: 22 │ │ │ Digest: DiBrBMshDiD9cThpaEgpcYSF76uV4hCoE1qRyQ3rnYCB │ │ └── │
╰──────────────────────────────────────────────────────────────────────────────────────────╯
```

In this example the object ID is 0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 . And the owner should be your account address. We achieved this by transferring the object to the sender in the last command of the transaction.

Another way to test that you have successfully created the list is to check the account objects.

```
bash $ sui client objects
```

It should have an object that looks similar to this:

```
bash ╭ ... ╮ │ ╭─────────────┬──────────────────────────────────────────────────────────────────────╮ │ │ │ objectId │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ │ version │ 22 │ │ │ │ digest │ /DUEiCLkaNSgzpZSq2vSV0auQQEQhyH9occq9grMBZM= │ │
│ │ objectType │ 0x468d..29fe::todo_list::TodoList │ │ │ ╰─────────────┴──────────────────────────────────────────────────────────────────────╯ │ │ ... │
```

The TodoList that we created in the previous step is an object that you can interact with as its owner. You can call functions defined in the todo_list module on this object. To demonstrate this, we will add an item to the list. First, we will add just one item, and in the second transaction we will add 3 and remove another one.

Double check that you have variables set up [from the previous step](), and then add one more variable for the list object.

```
bash $ export LIST_ID=0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553
```

Now we can construct the transaction to add an item to the list. The command will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Finish the Hello, Sui chapter'"
```

In this command, we are calling the add function in the todo_list package. The function takes two arguments: the list object and the item to add. The item is a string, so we need to wrap it in single quotes. The command will add the item to the list.

If everything is correct, you should see the output similar to the one we had in previous sections. Now you can check the list object to see if the item was added.

```
bash $ sui client object $LIST_ID
```

The output should contain the item that you have added.

```
bash ╭───────────────┬─────────────────────────────────────────────────────────────────────────────────────────╮ │ objectId │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ version │ 24 │ │ digest │ FGcXH8MGpMs5BdTnC62CQ3VLAwwexYg2id5DKU7Jr9aQ │ │ objType
│ 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ owner │
╭───────────────┬─────────────────────────────────────────────────────────────────────╮ │ │ │ │ AddressOwner │
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ │
╰───────────────┴─────────────────────────────────────────────────────────────────────╯ │ │ prevTx │ EJVK6FEHtfTdCuGkNsU1HcrmUBEN6H6jshfcptnw8Yt1 │ │
storageRebate │ 1558000 │ │ content │ ╭─────────────────┬──────────────────────────────────────────────────────────────────────────╮
│ │ │ │ dataType │ moveObject │ │ │ │ │ type │ 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ │ │
hasPublicTransfer │ true │ │ │ │ │ fields │ ╭───────┬──────────────────────────────────────────────────────────────────────╮ │ │ │ │ │ │ │ id │
╭────┬──────────────────────────────────────────────────────────────────╮ │ │ │ │ │ │ │ │ │ │ id │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ │ │ │ │ │ │ │
╰────┴──────────────────────────────────────────────────────────────────╯ │ │ │ │ │ │ │ │ items │ ╭───────────────────────────────────╮ │ │ │ │ │ │ │
│ │ │ finish the Hello, Sui chapter │ │ │ │ │ │ │ │ │ │ ╰───────────────────────────────────╯ │ │ │ │ │ │ │
╰──────┴────────────────────────────────────────────────────────────────────────────╯ │ │ │ │
╰──────────────────┴──────────────────────────────────────────────────────────────────────────────────╯ │
╰─────────────────┴──────────────────────────────────────────────────────────────────────────────────────────────╯
```

A JSON representation of the object can be obtained by adding the --json flag to the command.

```
bash $ sui client object $LIST_ID --json
```

```
bash { "objectId": "0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553", "version": "24", "digest":
"FGcXH8MGpMs5BdTnC62CQ3VLAwwexYg2id5DKU7Jr9aQ", "type": "0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList",
"owner": { "AddressOwner": "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" }, "previousTransaction":
"EJVK6FEHtfTdCuGkNsU1HcrmUBEN6H6jshfcptnw8Yt1", "storageRebate": "1558000", "content": { "dataType": "moveObject", "type":
"0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList", "hasPublicTransfer": true, "fields": { "id": { "id":
"0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553" }, "items": ["Finish the Hello, Sui chapter"] } } }
```

You can chain multiple commands in a single transaction. This shows the power of Transaction Blocks! Using the same list object, we will add three more items and remove one. The command will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Finish Concepts chapter'" \ --move-call
$PACKAGE_ID::todo_list::add @$LIST_ID "'Read the Move Basics chapter'" \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Learn about Object Model'" \
--move-call $PACKAGE_ID::todo_list::remove @$LIST_ID 0
```

If previous commands were successful, this one should not be any different. You can check the list object to see if the items were added and removed. The JSON representation is a bit more readable!

```
bash sui client object $LIST_ID --json
```

```
bash { "objectId": "0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553", "version": "25", "digest":
"EDTXDsteqPGAGu4zFAj5bbQGTkucWk4hhuUquk39enGA", "type": "0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList",
"owner": { "AddressOwner": "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" }, "previousTransaction":
"7SXLGBSh31jv8G7okQ9mEgnw5MnTfvzzHEHpWf3Sa9gY", "storageRebate": "1922800", "content": { "dataType": "moveObject", "type":
"0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList", "hasPublicTransfer": true, "fields": { "id": { "id":
"0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553" }, "items": [ "Finish Concepts chapter", "Read the Move Basics chapter", "Learn
about Object Model" ] } } }
```

Commands don't have to be in the same package or operate on the same object. Within a single transaction block, you can interact with multiple packages and objects. This is a powerful feature that allows you to build complex interactions on-chain!

In this guide, we have shown how to publish a package on the Move blockchain and interact with it using the Sui CLI. We have demonstrated how to create a new list object, add items to it, and remove them. We have also shown how to chain multiple commands in a single transaction block. This guide should give you a good starting point for building your own applications on the Sui blockchain!

## Sending Transactions

To demonstrate the interaction with the todo_list package, we will send a transaction to create a new list and add an item to it. Transactions are sent via the sui client ptb command, it allows using the [Transaction Blocks]() at full capacity. The command may look big and complex, but we go through it step by step.

Before we construct the command, let's store the values we will use in the transaction. Replace the 0x4.... with the address of the package you have published. And MY_ADDRESS variable will be automatically set to your address from the CLI output.

```
bash $ export PACKAGE_ID=0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe $ export MY_ADDRESS=$(sui client active-address)
```

Now to building an actual transaction. The transaction will consist of two parts: we will call the new function in the todo_list package to create a new list, and then we will transfer the list object to our account. The transaction will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --assign sender @$MY_ADDRESS \ --move-call $PACKAGE_ID::todo_list::new \ --assign list \ --transfer-
objects "[list]" sender
```

In this command, we are using the ptb subcommand to build a transaction. Parameters that follow it define the actual commands and actions that the transaction will perform. The first two calls we make are utility calls to set the sender address to the command inputs and set the gas budget for the transaction.

~~~bash

## sets the gas budget for the transaction

--gas-budget 100000000 \n

## registers a variable "sender=@..."

--assign sender @$MY_ADDRESS \n ```

Then we perform the actual call to a function in the package. We use the --move-call followed by the package ID, the module name, and the function name. In this case, we are calling the new function in the todo_list package.

```bash

## calls the "new" function in the "todo_list" package under the $PACKAGE_ID address

--move-call $PACKAGE_ID::todo_list::new ```

The function that we defined actually returns a value, which we want need to store. We use the --assign command to give a name to the returned value. In this case, we are calling it list . And then we transfer the object to our account.

```bash --move-call $PACKAGE_ID::todo_list::new \

## assigns the result of the "new" function to the "list" variable (from the previous step)

--assign list \

## transfers the object to the sender

--transfer-objects "[list]" sender ```

Once the command is constructed, you can run it in the terminal. If everything is correct, you should see the output similar to the one we had in previous sections. The output will contain the transaction digest, the transaction data, and the transaction effects.

```bash Transaction Digest: BJwYEnuuMzU4Y8cTwMoJbbQA6cLwPmwxvsRpSmvThoK8

~~~
╭──────────────────────────────────────────────────────────────────────────────╮
│ Transaction Data │
├──────────────────────────────────────────────────────────────────────────────┤
│ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas Owner: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ Gas
Budget: 100000000 MIST │ │ Gas Price: 1000 MIST │ │ Gas Payment: │ │ ┌── │ │ │ ID: 0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Version: 22
│ │ │ Digest: DiBrBMshDiD9cThpaEgpcYSF76uV4hCoE1qRyQ3rnYCB │ │ └── │ │ │ │ Transaction Kind: Programmable │ │
╭──────────────────────────────────────────────────────────────────────────╮ │ │
│ Input Objects │ │ │
├──────────────────────────────────────────────────────────────────────────┤ │ │
│ 0 Pure Arg: Type: address, Value: "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" │ │ │
╰──────────────────────────────────────────────────────────────────────────╯ │ │
╭─────────────────────────────────────────────────────────────╮ │ │ │ Commands │ │ │
├─────────────────────────────────────────────────────────────┤ │ │ │ 0 MoveCall: │ │ │ │ ┌ │ │ │ │ │
Function: new │ │ │ │ │ Module: todo_list │ │ │ │ │ Package: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe │ │ │ │ └ │ │ │ │ │ │ │ │ │ 1 TransferObjects:
│ │ │ │ ┌ │ │ │ │ │ Arguments: │ │ │ │ │ Result 0 │ │ │ │ │ Address: Input 0 │ │ │ │ └ │ │ │
╰─────────────────────────────────────────────────────────────╯ │ │ │ │ Signatures: │ │
C5Lie4dtP5d3OkKzFBa+xM0BiNoB/A4ItthDCRTRBUrEE+jXeNs7mP4AuGwi3nzfTskh29+R1j1Kba4Wdy3QDA== │ │ │
╰──────────────────────────────────────────────────────────────────────────────╯

╭──────────────────────────────────────────────────────────────────────────╮ │ Transaction
Effects │ ├──────────────────────────────────────────────────────────────────────────────┤ │
Digest: BJwYEnuuMzU4Y8cTwMoJbbQA6cLwPmwxvsRpSmvThoK8 │ │ Status: Success │ │ Executed Epoch: 1213 │ │ │ │ Created Objects: │ │ ┌── │ │ │ ID:
0x74973c4ea2e78dc409f60481e23761cee68a48156df93a93fbcceb77d1cacdf6 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │
│ │ Version: 23 │ │ │ Digest: DuHTozDHMsuA7cFnWRQ1Gb8FQghAEBaj3inasJxqYq1c │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 )
│ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │ │ Gas Object: │ │ ┌── │ │ │ ID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 )
│ │ │ Version: 23 │ │ │ Digest: 82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │ │ Gas Cost Summary: │ │ Storage Cost: 2318000 MIST │ │ Computation Cost:
1000000 MIST │ │ Storage Rebate: 978120 MIST │ │ Non-refundable Storage Fee: 9880 MIST │ │ │ │ Transaction Dependencies: │ │
FSz2fYXmKqTf77mFXNq5JK7cKY8agWja7V5yDKEgL8c3 │ │ GgMZKTt482DYApbAZkPDtdssGHZLbxgjm2uMXhzJax8Q │
╰──────────────────────────────────────────────────────────────────────────╯

╭───────────────────────────────╮ │ No transaction block events │ ╰───────────────────────────────╯

╭──────────────────────────────────────────────────────────────────────────╮ │ Object
Changes │
├──────────────────────────────────────────────────────────────────────────┤ │ Created
Objects: │ │ ┌── │ │ │ ObjectID: 0x74973c4ea2e78dc409f60481e23761cee68a48156df93a93fbcceb77d1cacdf6 │ │ │ Sender:
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │
│ │ ObjectType: 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ Version: 23 │ │ │ Digest:
DuHTozDHMsuA7cFnWRQ1Gb8FQghAEBaj3inasJxqYq1c │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ Owner:
Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │ Version: 23 │ │ │ Digest:
82fwKarGuDhtomr5oS6ZGNvZNw9QVXLSbPdQu6jQgNV7 │ │ └── │
╰──────────────────────────────────────────────────────────────────────────╯

╭──────────────────────────────────────────────────────────────────────────╮ │ Balance
Changes │ ├──────────────────────────────────────────────────────────────────────────┤ │
┌── │ │ │ Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ CoinType: 0x2::sui::SUI │ │ │ Amount: -2339880 │ │ └── │
╰──────────────────────────────────────────────────────────────────────────╯ ```
```

The section that we want to focus on is the "Object Changes". More specifically, the "Created Objects" part of it. It contains the object ID, the type and the version of the TodoList that you have created. We will use this object ID to interact with the list.

```
bash ╭──────────────────────────────────────────────────────────────╮ │ Object Changes │
├──────────────────────────────────────────────────────────────┤ │ Created Objects: │ │ ┌── │ │ │ ObjectID:
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType:
```
```

```
0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ Version: 22 │ │ │ Digest:
HyWdUpjuhjLY38dLpg6KPHQ3bt4BqQAbdF5gB8HQdEqG │ │ └── │ │ Mutated Objects: │ │ ┌── │ │ │ ObjectID:
0xe5ddeb874a8d7ead328e9f2dd2ad8d25383ab40781a5f1aefa75600973b02bc4 │ │ │ Sender: 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │
Owner: Account Address ( 0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 ) │ │ │ ObjectType: 0x2::coin::Coin<0x2::sui::SUI> │ │ │
Version: 22 │ │ │ Digest: DiBrBMshDiD9cThpaEgpcYSF76uV4hCoE1qRyQ3rnYCB │ │ └── │
╰──────────────────────────────────────────────────────────────────────────────────────────╯
```

In this example the object ID is 0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 . And the owner should be your account address. We achieved this by transferring the object to the sender in the last command of the transaction.

Another way to test that you have successfully created the list is to check the account objects.

```
bash $ sui client objects
```

It should have an object that looks similar to this:

```
bash ╭ ... ╮ │ ╭────────────┬──────────────────────────────────────────────────────────────────╮ │ │ │ objectId │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ │ version │ 22 │ │ │ │ digest │ /DUEiCLkaNSgzpZSq2vSV0auQQEQhyH9occq9grMBZM= │ │
│ │ objectType │ 0x468d..29fe::todo_list::TodoList │ │ │ ╰────────────┴──────────────────────────────────────────────────────────────────╯ │ │ ... │
```

The TodoList that we created in the previous step is an object that you can interact with as its owner. You can call functions defined in the todo_list module on this object. To demonstrate this, we will add an item to the list. First, we will add just one item, and in the second transaction we will add 3 and remove another one.

Double check that you have variables set up [from the previous step](#) , and then add one more variable for the list object.

```
bash $ export LIST_ID=0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553
```

Now we can construct the transaction to add an item to the list. The command will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Finish the Hello, Sui chapter'"
```

In this command, we are calling the add function in the todo_list package. The function takes two arguments: the list object and the item to add. The item is a string, so we need to wrap it in single quotes. The command will add the item to the list.

If everything is correct, you should see the output similar to the one we had in previous sections. Now you can check the list object to see if the item was added.

```
bash $ sui client object $LIST_ID
```

The output should contain the item that you have added.

```
bash ╭───────────────┬─────────────────────────────────────────────────────────────────────────────────────╮ │ objectId │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ version │ 24 │ │ digest │ FGcXH8MGpMs5BdTnC62CQ3VLAwwexYg2id5DKU7Jr9aQ │ │ objType
│ 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ owner │
╭────────┬─────────────────────────────────────────────────────────────────╮ │ │ │ │ AddressOwner │
0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1 │ │ │ │
╰────────┴─────────────────────────────────────────────────────────────────╯ │ │ prevTx │ EJVK6FEHtfTdCuGkNsU1HcrmUBEN6H6jshfcptnw8Yt1 │ │
storageRebate │ 1558000 │ │ content │ ╭───────────────┬──────────────────────────────────────────────────────────────╮
│ │ │ │ dataType │ moveObject │ │ │ │ │ type │ 0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList │ │ │ │ │
hasPublicTransfer │ true │ │ │ │ │ fields │ ╭─────┬────────────────────────────────────────────────╮ │ │ │ │ │ │ │ id │
╭───┬──────────────────────────────────────────────────────────────────╮ │ │ │ │ │ │ │ │ │ │ id │
0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553 │ │ │ │ │ │ │ │ │ │
╰───┴──────────────────────────────────────────────────────────────────╯ │ │ │ │ │ │ │ │ items │ ╭─────────────────────────────╮ │ │ │ │ │ │ │
│ │ │ finish the Hello, Sui chapter │ │ │ │ │ │ │ │ │ │ ╰─────────────────────────────╯ │ │ │ │ │ │ │
╰───┴─────────────────────────────────────────────────────────────────────╯ │ │ │ │
╰──────────────┴─────────────────────────────────────────────────────────────────────────────╯ │
╰───────────┴───────────────────────────────────────────────────────────────────────────────────────────╯
```

A JSON representation of the object can be obtained by adding the --json flag to the command.

```
bash $ sui client object $LIST_ID --json
```

```
bash { "objectId": "0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553", "version": "24", "digest":
"FGcXH8MGpMs5BdTnC62CQ3VLAwwexYg2id5DKU7Jr9aQ", "type": "0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList",
"owner": { "AddressOwner": "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" }, "previousTransaction":
"EJVK6FEHtfTdCuGkNsU1HcrmUBEN6H6jshfcptnw8Yt1", "storageRebate": "1558000", "content": { "dataType": "moveObject", "type":
"0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList", "hasPublicTransfer": true, "fields": { "id": { "id":
"0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553" }, "items": ["Finish the Hello, Sui chapter"] } } }
```

You can chain multiple commands in a single transaction. This shows the power of Transaction Blocks! Using the same list object, we will add three more items and remove one. The command will look like this:

```
bash $ sui client ptb \ --gas-budget 100000000 \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Finish Concepts chapter'" \ --move-call
$PACKAGE_ID::todo_list::add @$LIST_ID "'Read the Move Basics chapter'" \ --move-call $PACKAGE_ID::todo_list::add @$LIST_ID "'Learn about Object Model'" \
--move-call $PACKAGE_ID::todo_list::remove @$LIST_ID 0
```

If previous commands were successful, this one should not be any different. You can check the list object to see if the items were added and removed. The JSON representation is a bit more readable!

```
bash sui client object $LIST_ID --json
```

```
bash { "objectId": "0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553", "version": "25", "digest":
"EDTXDsteqPGAGu4zFAj5bbQGTkucWk4hhuUquk39enGA", "type": "0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList",
"owner": { "AddressOwner": "0x091ef55506ad814920adcef32045f9078f2f6e9a72f4cf253a1e6274157380a1" }, "previousTransaction":
"7SXLGBSh31jv8G7okQ9mEgnw5MnTfvzzHEHpWf3Sa9gY", "storageRebate": "1922800", "content": { "dataType": "moveObject", "type":
"0x468daa33dfcb3e17162bbc8928f6ec73744bb08d838d1b6eb94eac99269b29fe::todo_list::TodoList", "hasPublicTransfer": true, "fields": { "id": { "id":
"0x20e0bede16de8a728ab25e228816b9059b45ebea49c8ad384e044580b2d3e553" }, "items": [ "Finish Concepts chapter", "Read the Move Basics chapter", "Learn
about Object Model" ] } } }
```

Commands don't have to be in the same package or operate on the same object. Within a single transaction block, you can interact with multiple packages and objects. This is a powerful feature that allows you to build complex interactions on-chain!

In this guide, we have shown how to publish a package on the Move blockchain and interact with it using the Sui CLI. We have demonstrated how to create a new list object, add items to it, and remove them. We have also shown how to chain multiple commands in a single transaction block. This guide should give you a good starting point for building your own applications on the Sui blockchain!

## Conclusion

In this guide, we have shown how to publish a package on the Move blockchain and interact with it using the Sui CLI. We have demonstrated how to create a new list object, add items to it, and remove them. We have also shown how to chain multiple commands in a single transaction block. This guide should give you a good starting point for building your own applications on the Sui blockchain!