

# Gnoland Transaction Types

---

## Understanding `call`, `run`, `addpkg` and more




---

How to interact with the Gno blockchain



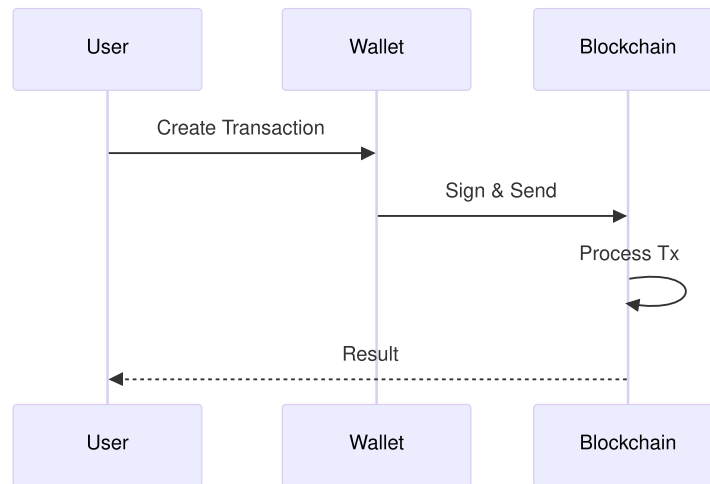
# What are Transactions?

## Blockchain Interactions

-  Signed operations
-  Require gas fees
-  Change blockchain state







## Key Components:


1. Sender (signer)
2. Action ( `call` , `run` , etc.)
3. Arguments
4. Gas fee





# Transaction Types Overview

Type	Command	Purpose	State Change
 <b>Call</b>	<code>maketx call</code>	Execute specific function	✅ Permanent
 <b>Run</b>	<code>maketx run</code>	Execute entire script	❌ Temporary
 <b>AddPkg</b>	<code>maketx addpkg</code>	Deploy new contract	✅ Permanent
 <b>Send</b>	<code>maketx send</code>	Transfer coins	✅ Permanent
 <b>Exec</b>	<code>maketx exec</code>	Run local file	❌ Temporary
 <b>Query</b>	<code>query</code>	Read state	❌ Read-only

 **Key Difference:** Permanent transactions change blockchain state, temporary transactions don't persist



# Call Transactions

## Execute Specific Functions

```
gnokey maketx call \  
  --pkgpath "gno.land/r/demo/boards" \  
  --func "CreateThread" \  
  --args "Crypto" "Market News" "BTC up 10%" \  
  --gas-fee 1gnot
```

✓ **Use Cases** - Modify contract state -  
Payable functions - Gas-efficient

⚠ **Limitations** - Must know function signature  
- Package must be deployed



# Run Transactions

## Execute Complete Scripts

```
gnokey maketx run \  
  --pkgpath "gno.land/r/demo/my_script" \  
  --gas-fee 2gnot \  
  --broadcast
```

✓ **Use Cases** - One-time computations -  
Script-like operations - Initialize contracts

⚠ **Limitations** - No persistent state - Higher  
gas cost



# AddPkg Transactions

## Deploy New Contracts







```
gnokey maketx addpkg \  
  --pkgpath "gno.land/r/myapp/mytoken" \  
  --deposit "1000gnot" \  
  --pkgdir "./mytoken"
```

✓ **Use Cases** - Deploy new realms - Update existing packages - Store contract code

⚠ **Limitations** - Requires deposit - Complex deployment

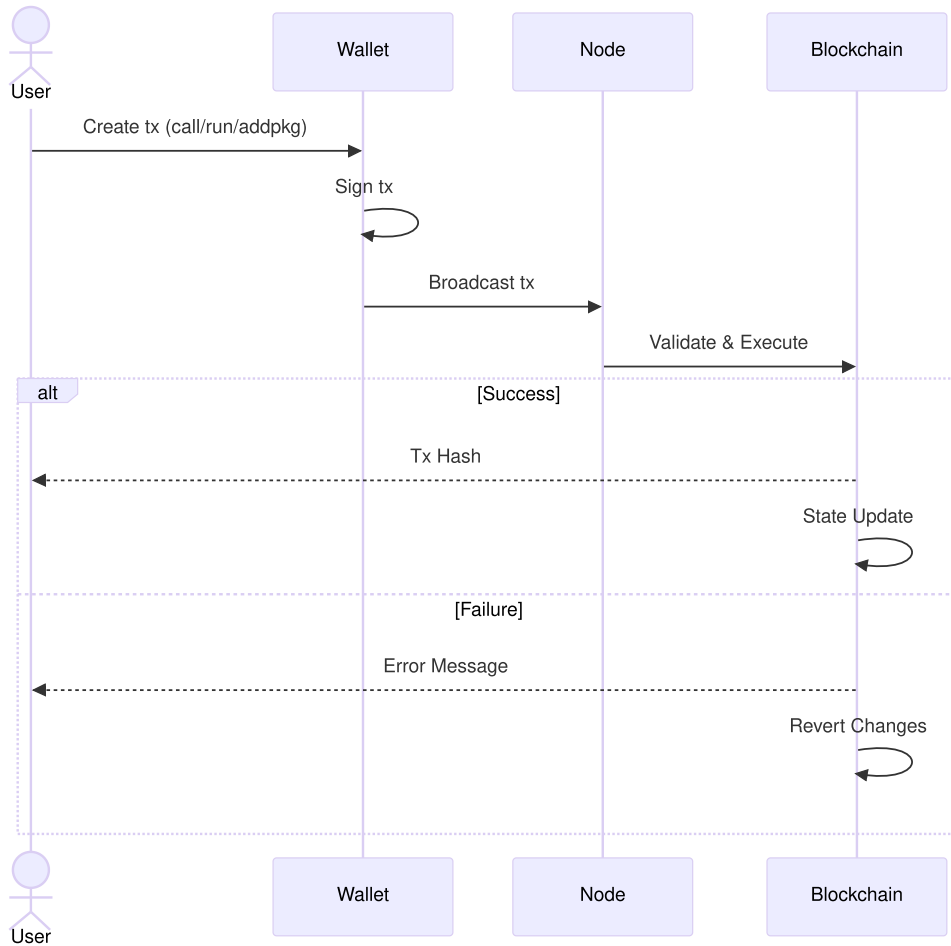


# Transaction Comparison

Feature	 Call	 Run	 AddPkg
State Change		 (Temporary)	
Gas Cost	Low	Medium	High
Persistence	Permanent	Ephemeral	Permanent
Deployment	Not required	Not required	Required
Best For	Contract methods	One-off scripts	Deploying code



# Transaction Lifecycle







# Practical Examples

## 1. Deploy a Token Contract

```
gnokey maketx addpkg \  
  --pkgpath "gno.land/r/mydapp/token" \  
  --pkgdir "./token-contract"
```

## 2. Mint New Tokens

```
gnokey maketx call \  
  --pkgpath "gno.land/r/mydapp/token" \  
  --func "Mint" \  
  --args "g1myaddress" "1000000"
```

## 3. Run Setup Script

```
gnokey maketx run \  
  --pkgpath "gno.land/r/mydapp/setup" \  
  --gas-fee 3gnot
```



# Common Errors & Fixes

## ✗ "out of gas"

```
# Solution:  
Increase gas:  
--gas-wanted 2000000 \  
--gas-fee 5gnot
```

## ✗ "invalid pkgpath"

```
# Solution:  
Verify exact path:  
gno.land/r/demo/boards
```

## ✗ "function not found"

```
# Solution:  
Check function name:  
--func "CreateBoard"
```

## ✗ "insufficient deposit"

```
# Solution:  
Add more funds:  
--deposit "2000gnot"
```



# Pro Tips

## 1. Dry Run First

```
gnokey maketx call ... --dry-run
```

## 2. Estimate Gas

```
gnokey query estimate-gas --tx <tx-file>
```

## 3. View Tx History

```
gnokey query txs --tags call
```

## 4. Test Locally

```
gnokey maketx run --pkgpath localfile.gno
```



# Resources & Next Steps

## Official Documentation

[gno.land Transactions Guide](#)

## Next Steps:

1. Try all transaction types
2. Build a full dApp workflow
3. Explore advanced gas management
4. Implement transaction batching

```
# View help for all commands  
gnokey maketx --help
```



**Master transactions to unlock Gnoland's full potential!**