



# GovDAO

---

## Decentralized Governance on Gno.land

---

# What is a DAO?

**DAO** = **D**ecentralized **A**utonomous **O**rganization

- **No central authority**
- **Members vote** on **Proposal**
- Manual execution through **Smart contracts**

## Two Types:

- **Permissioned** - Only invited members can participate
- **Permissionless** - Anyone can join and vote through public procedure



# What is GovDAO?

**GovDAO** is the **Governance DAO** of **Gno.land**

- It is a **permissioned** DAO with **3 tiers** optional based voting
- Built with a **proxy pattern** for upgradeability and safety

## How it works:

1. 📝 DAO's members submit **proposals** through the proxy
2. 🗳️ Members **vote** based on their tier privileges
3. 🚀 Approved proposals can be executed with **supermajority** (66%)
  - All voters must have **registered namespaces**



# Three-Tier Membership System

**Tier-based voting is optional** - depends on proposal's filter configuration

**T1**

**Highest Tier**

Vote on Everything

**T2**

**Mid Tier**

Vote on T2 & T3

**T3**

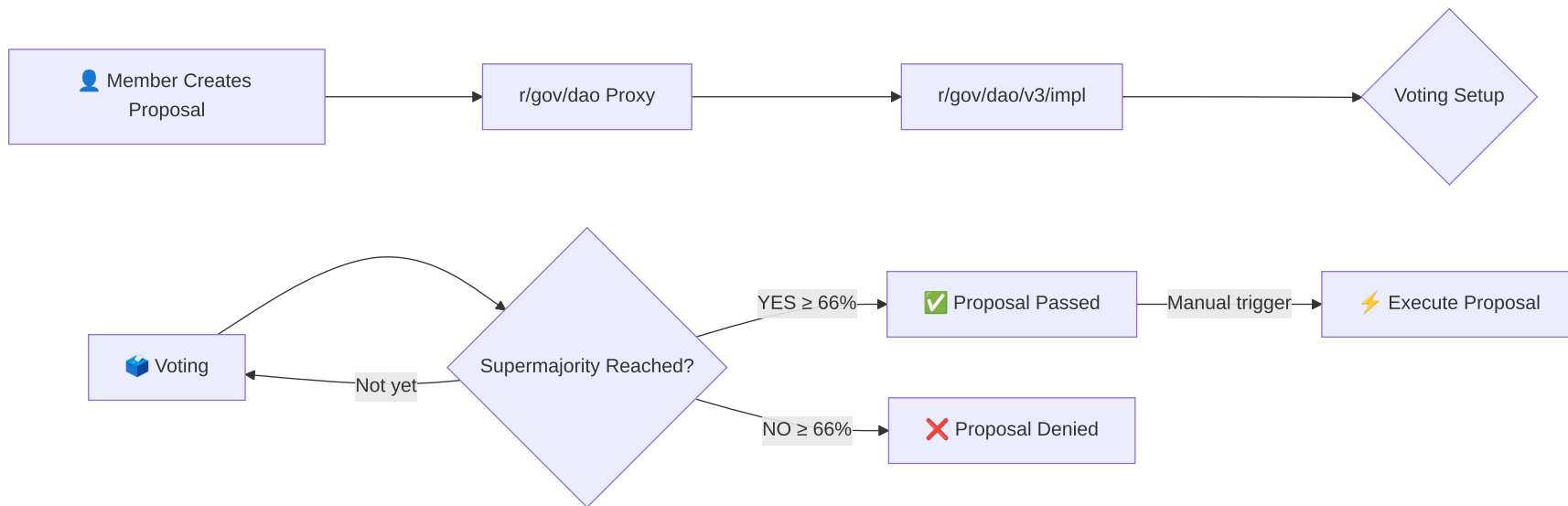
**Base Tier**

Vote on T3 Only

Tier assignment and invitation points is determined at genesis



# GovDAO Flow: Creation & Voting





# How to Create a Proposal

## Step 1: Prepare Your Executor

```
// Simple Executor - basic execution
executor := dao.NewSimpleExecutor(
    func(realm) error {
        // Your code to execute if proposal passes
        return nil
    },
    "Description of what this will do",
)

// OR Safe Executor - only allowed DAOs can execute (Not usable in v3)
safeExecutor := dao.NewSafeExecutor(executor)
```



# How to Create a Proposal

## Step 2: Create the Request

```
// Basic request
request := dao.NewProposalRequest(
    "Proposal Title",
    "Detailed description...",
    executor,
)

// OR with filter (for tier-based voting)
request := dao.NewProposalRequestWithFilter(
    "Add T1 Member",
    "Proposal to add new T1 member...",
    executor,
    FilterByTier{Tier: memberstore.T1}, // Only T1 can vote
)
```



# How to Create a Proposal

## Step 3: Submit (Members Only!)

```
proposalID, err := dao.CreateProposal(request)
```





# How to Vote on a Proposal

Find the proposal ID & submit your vote:

```
// Simple CLI voting
dao.MustVoteOnProposalSimple(proposalID, "YES") // or "NO", "ABSTAIN"
```

```
# Example gnokey command:
gnokey maketx call \
  -pkgpath "gno.land/r/gov/dao" \
  -func "MustVoteOnProposalSimple" \
  -args "123" \
  -args "YES" \
  -gas-fee "1000000ugnot" \
  -gas-wanted "2000000" \
  -broadcast \
  -chainid "staging" \
  -remote "https://rpc.test9.testnets.gno.land:443" \
  yourkeyname
```



# How to Vote on a Proposal

## Requirements:

- Must be a member
- Must have a namespace (identity)
- Your tier must be allowed for this proposal
- One vote per member
- Proposal must still be open

# What can GovDAO vote on?

- Treasury Operations
- Member Management
- Protocol Parameters
- Governance Rules
- System Configuration



# Execution Through Proposals

**Any programmable action** within Gno can be voted on!

## Example: Treasury Payment

```
executor := dao.NewSafeExecutor(  
    dao.NewSimpleExecutor(  
        func(cur realm) error {  
            return treasury.Send(cross, treasury.Payment{  
                BankerID: "coins",  
                To: "g1developer...",  
                Amount: "1000000ugnot",  
                Memo: "Development grant",  
            })  
        },  
        "Pay developer 1000 GNOT",  
    ),  
)
```

# DAOs in Blockchain

## Examples:

- **The DAO (2016)** - First major experiment on Ethereum
- **MakerDAO** - Decentralized finance and stablecoin governance
- **Uniswap DAO** - Decentralized exchange governance