An Overview of Substrate

Web3 Foundation | https://web3.foundation | @web3foundation

Parity Technologies | https://parity.io | @paritytech

Bill Laboon
Technical Education Lead at Web3 Foundation

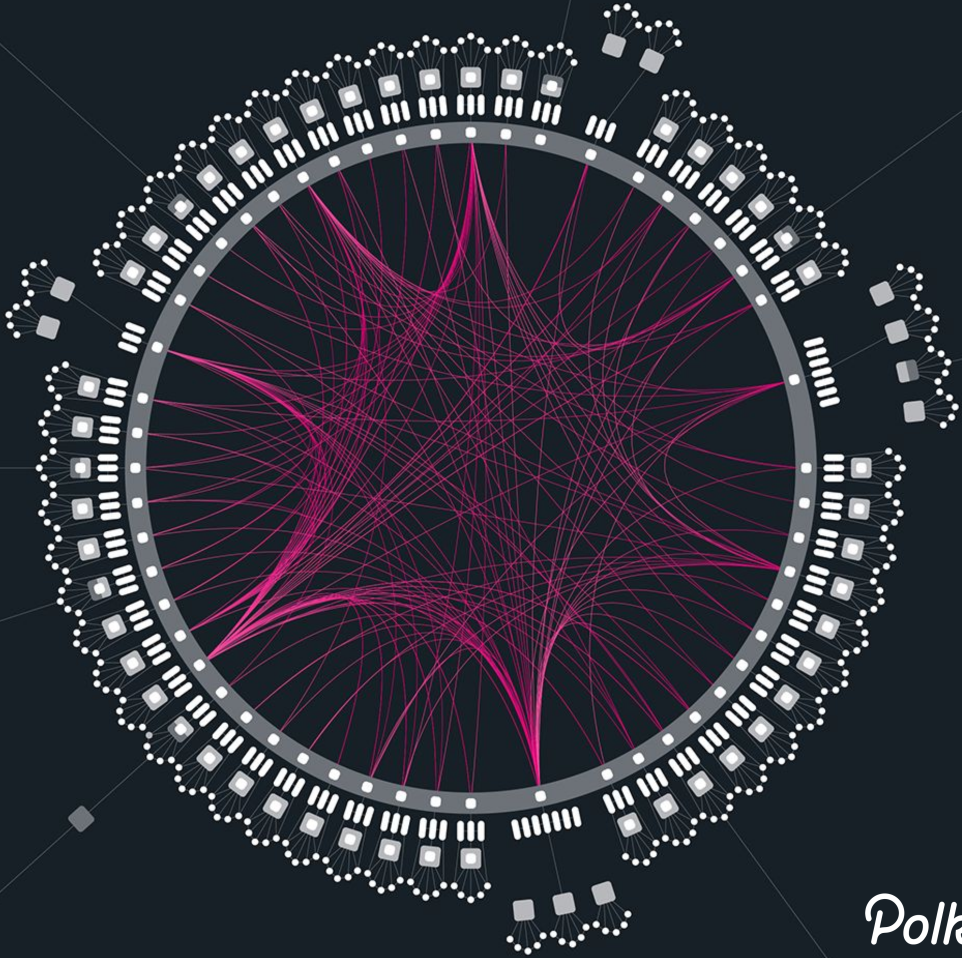
Twitter: @BillLaboon
Email: bill@web3.foundation

Bill Laboon is the Technical Education Lead at the Web3 Foundation. Before this, he was a lecturer in the Computer Science Department of the University of Pittsburgh, teaching courses in software quality assurance, software engineering, and blockchain technology. He is a frequent speaker at conferences on a variety of topics, including cryptocurrency, software quality, and the ethics of software development. He is the author of two books: A Friendly Introduction to Software Testing, an undergraduate textbook; and Strength in Numbers, a near-future novel set in a world in which cryptocurrency has eliminated traditional money. Bill has a BS in Computer Science and Political Science from the University of Pittsburgh, as well as an MS in Software Design & Management from Carnegie Mellon University.

web3
foundation

Web3 Foundation is a Switzerland-based foundation dedicated to "nurturing and stewarding cutting-edge technologies and applications in the fields of cryptographically-powered decentralized software protocols."

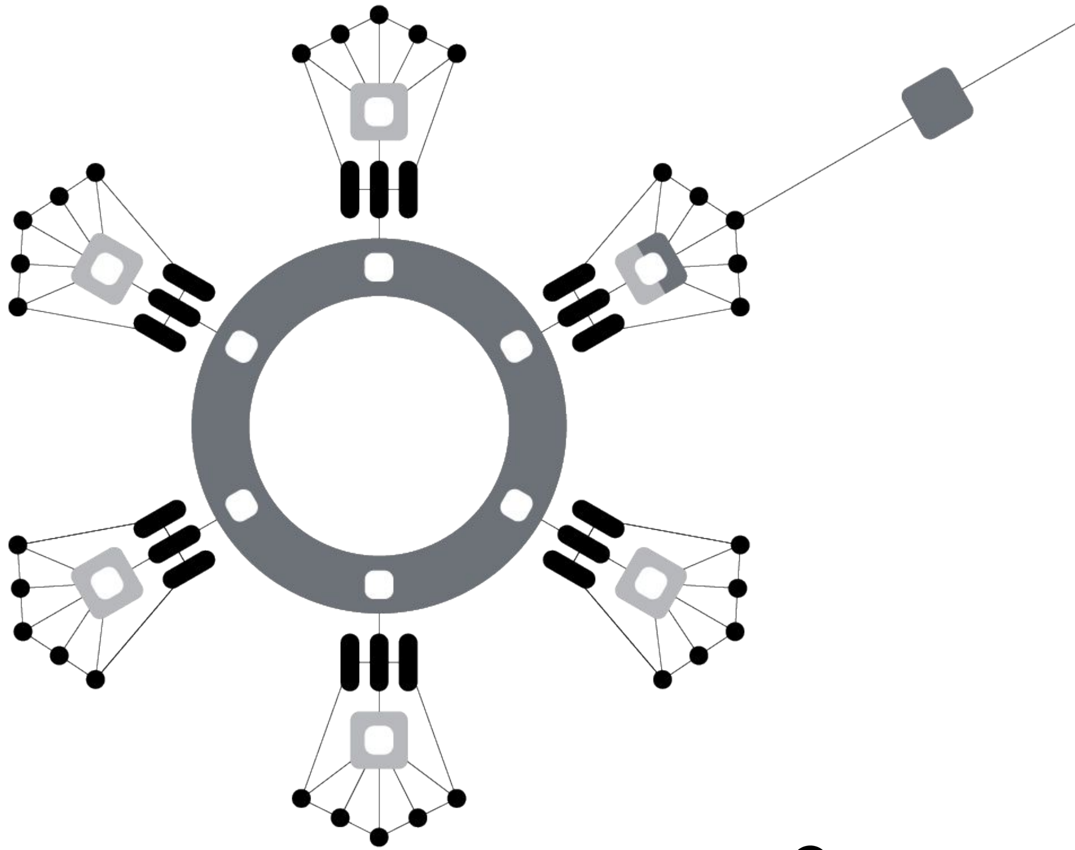
POLKADOT BASICS



Polkadot.

INTRODUCTION
Polkadot

▬	Validators
●	Collators
▩	Relaychain
▣	Parachains
■	Parathreads
▣	Bridges
■	Other Blockchains



Polkadot.

INTRODUCTION

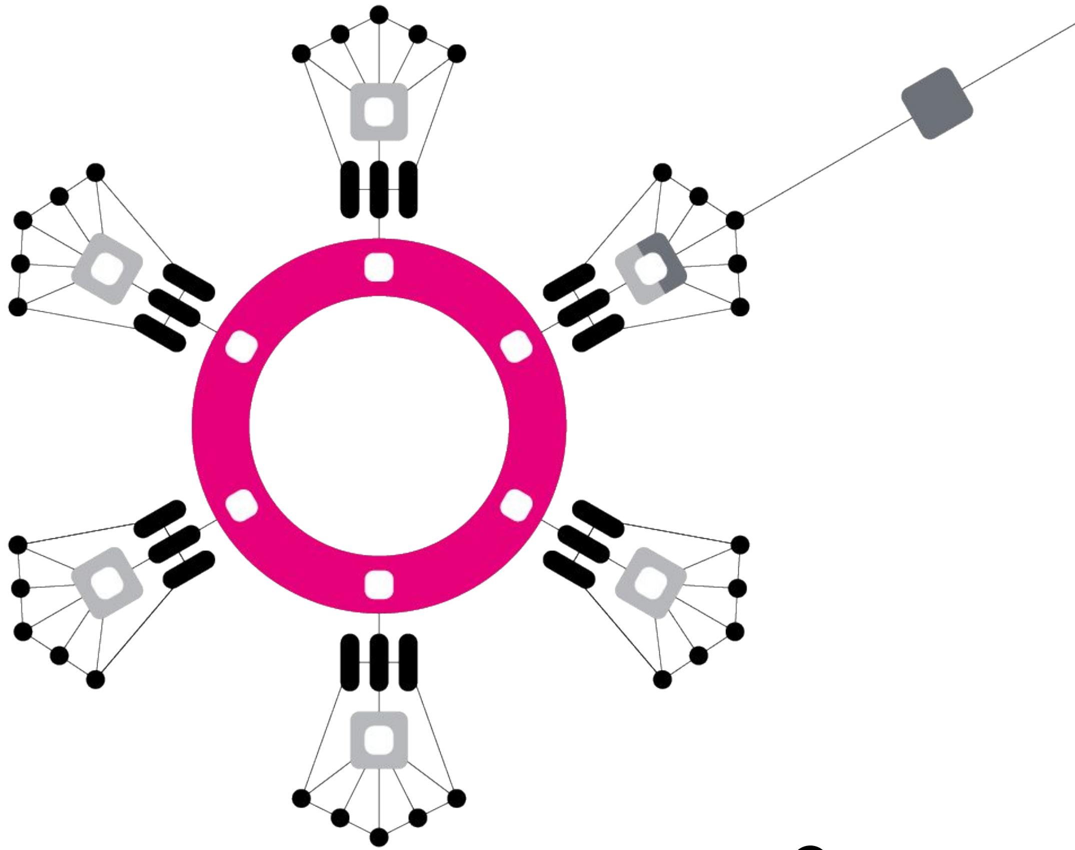
Relay Chain

The relay chain is the main chain of Polkadot.

- *Other connected "parachains" are heterogenous shards - blockchains which share security and communicate with each other*

- *Relay chain holds the states of the parachains.*

- *It is secured via nominated proof of stake*



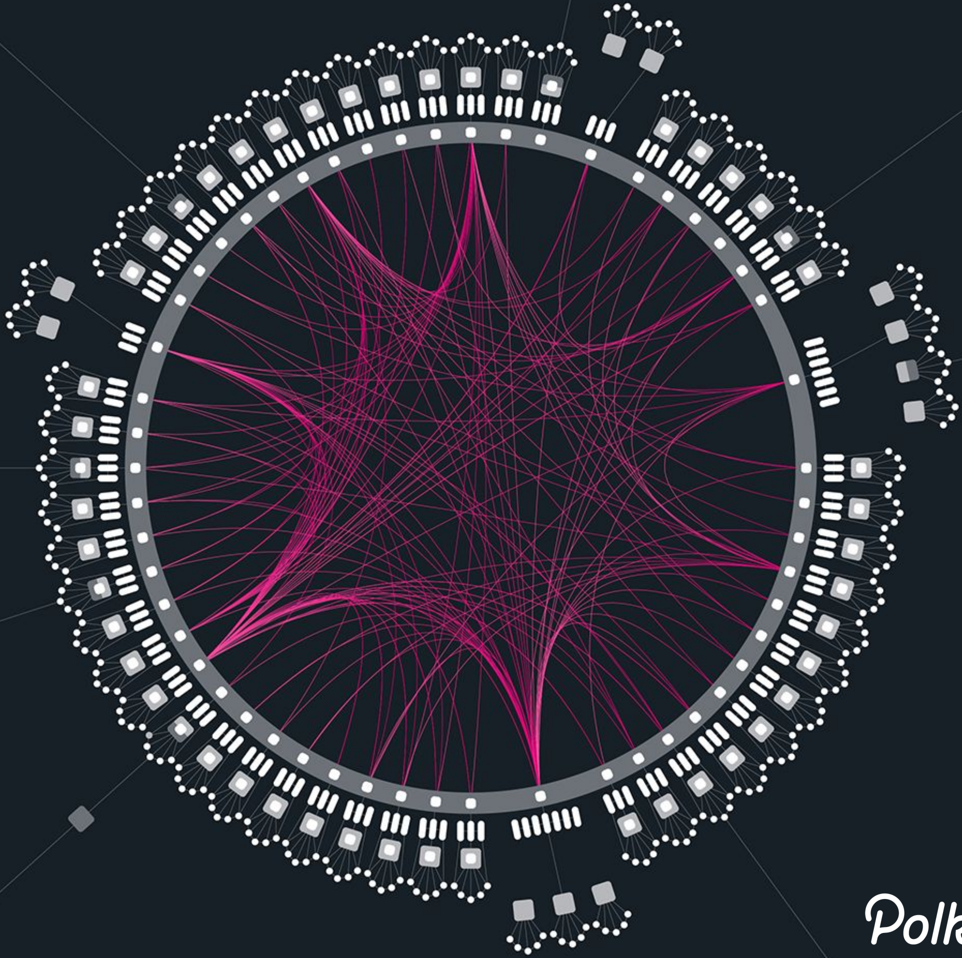
Polkadot.

WHY POLKADOT?

- Heterogenous shards - develop chains to your specification
- Shared security across all parachains
- Cross-chain communication built in to the protocol
- Nominated proof-of-stake
- Thought-through, on-chain governance
- Large and growing ecosystem
- Substrate!

Polkadot.

SUBSTRATE BASICS



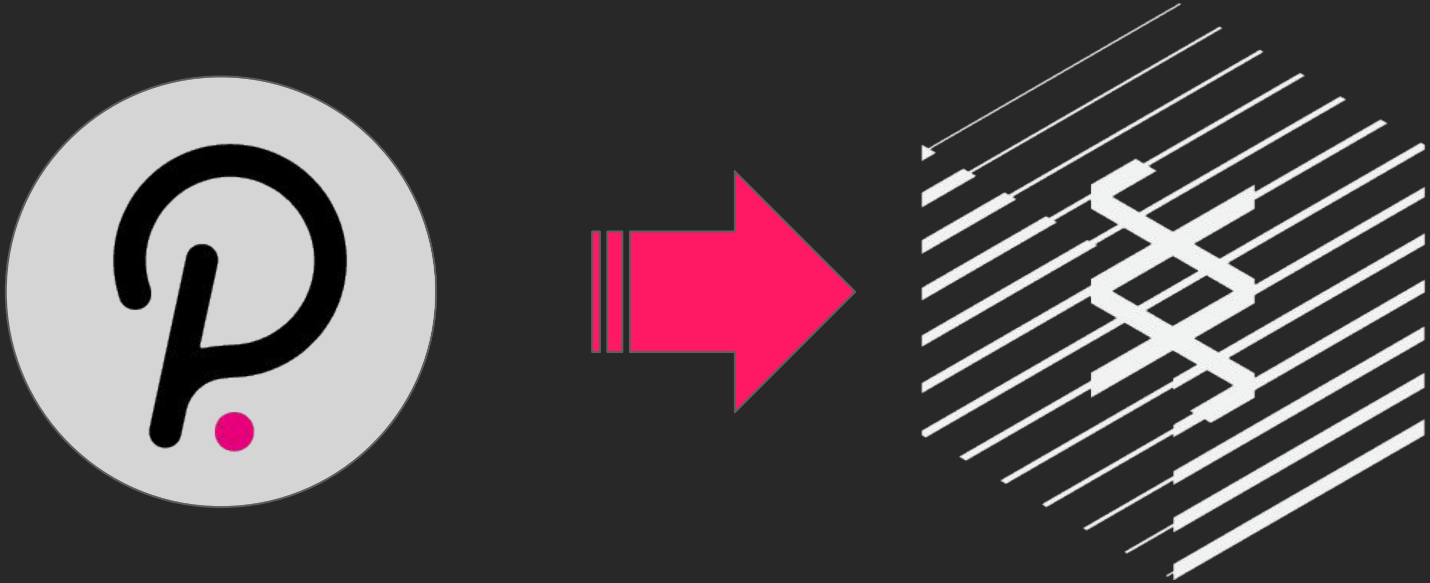
Parity has a lot of blockchain building experience...

github.com/paritytech/
parity-ethereum

github.com/paritytech/
parity-bitcoin

github.com/paritytech/
polkadot

From Polkadot, came Substrate.



What is Substrate?

Substrate is an **open source**, **modular**, and **extensible** framework for building blockchains.

What is Substrate?

Substrate provides all the core components of a Blockchain:

- Database Layer
- Networking Layer
- Consensus Engine
- Transaction Queue
- Library of Runtime Modules

Each of which can be customized and extended.

Why Substrate?

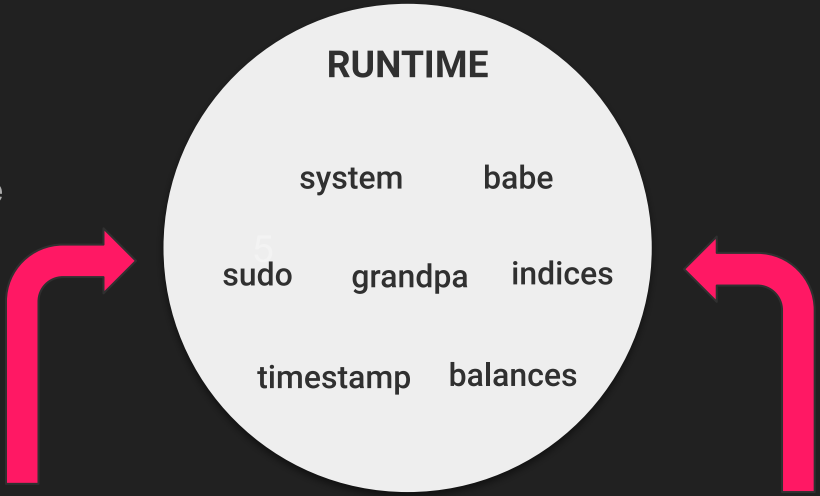
- Automatically become a parachain or parathread
- Or run as an independent blockchain
- Benefit from features added to Substrate in the future
- Share modules (pallets) developed by you or others

The Substrate Runtime

The runtime is the **block execution logic** of the blockchain, a.k.a. the State Transition Function.

It is composed of

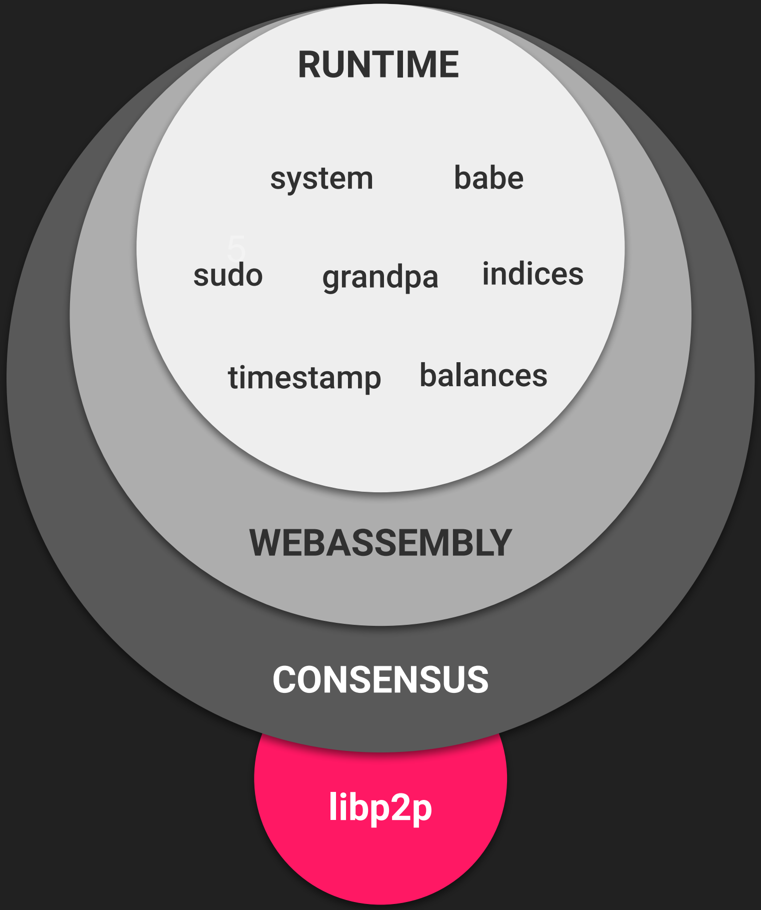
FRAME Pallets.



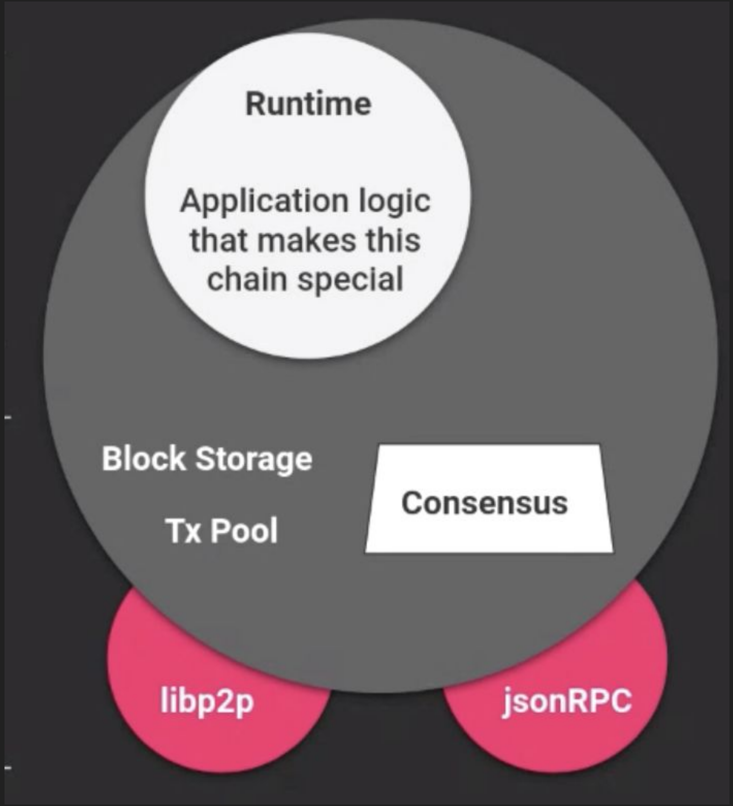
Pallets built with FRAME			
assets	babe	balances	collective
contract	democracy	elections	grandpa
indices	grandpa	indices	membership
offences	session	staking	sudo
system	timestamp	treasury	and more...

Substrate Node Template

- A working Substrate node.
- Basic cryptocurrency chain with administrative governance.
- Easily add and remove pallets built with FRAME.
- Create your own modules to customize your chain functionality.

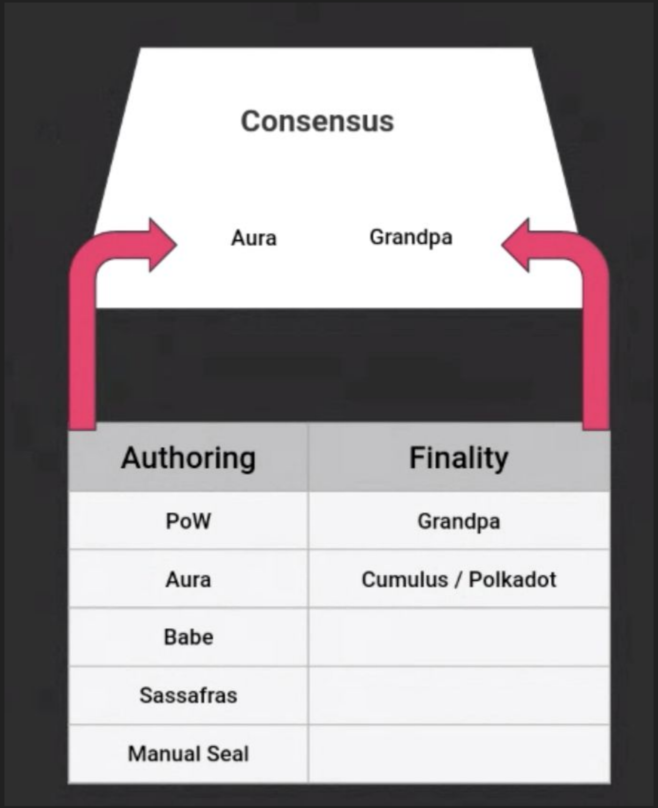


Architecture of a Node



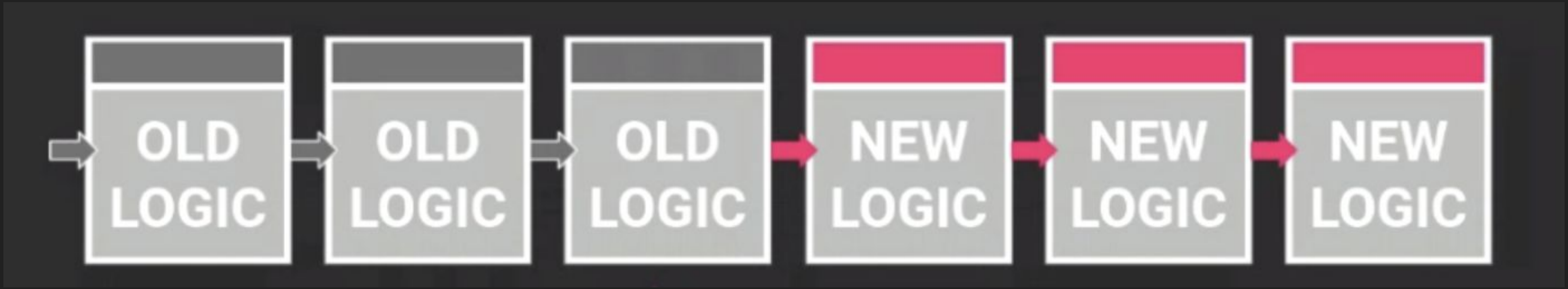
Consensus

1. Who can author blocks?
2. When are blocks considered final?



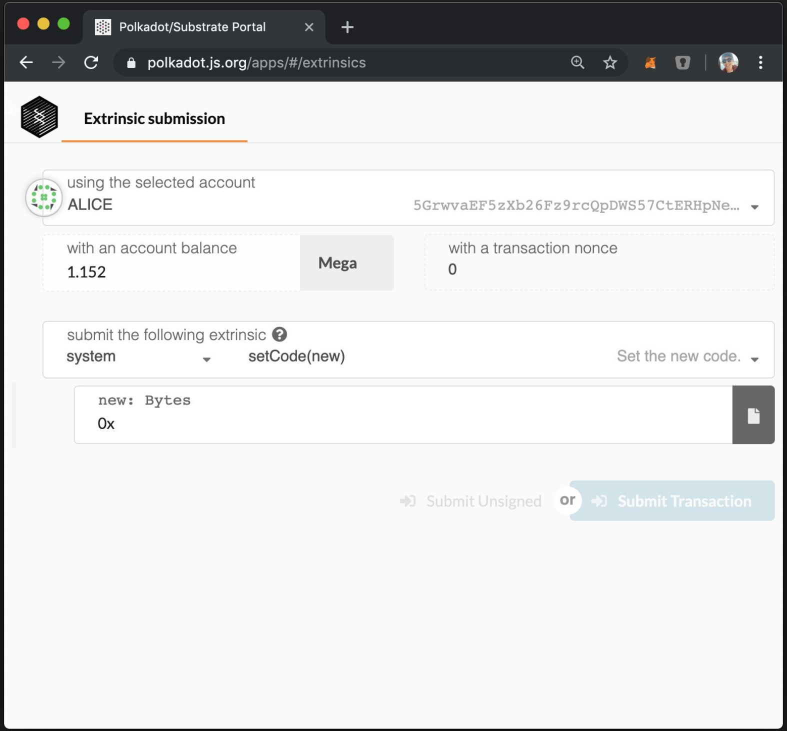
Authoring	Finality
PoW	Grandpa
Aura	Cumulus / Polkadot
Babe	
Sassafras	
Manual Seal	

Forkless Upgrades



Polkadot JS Apps

- Generalized and hosted UI
- Quickly test new functionality
- Loaded with general tools like:
  - Creating transactions
  - Read storage
  - See events
  - and way more…
- Great for development



using the selected account
ALICE 5GrwvaEF5zXb26Fz9rcQpDWS57CtERHpNe…

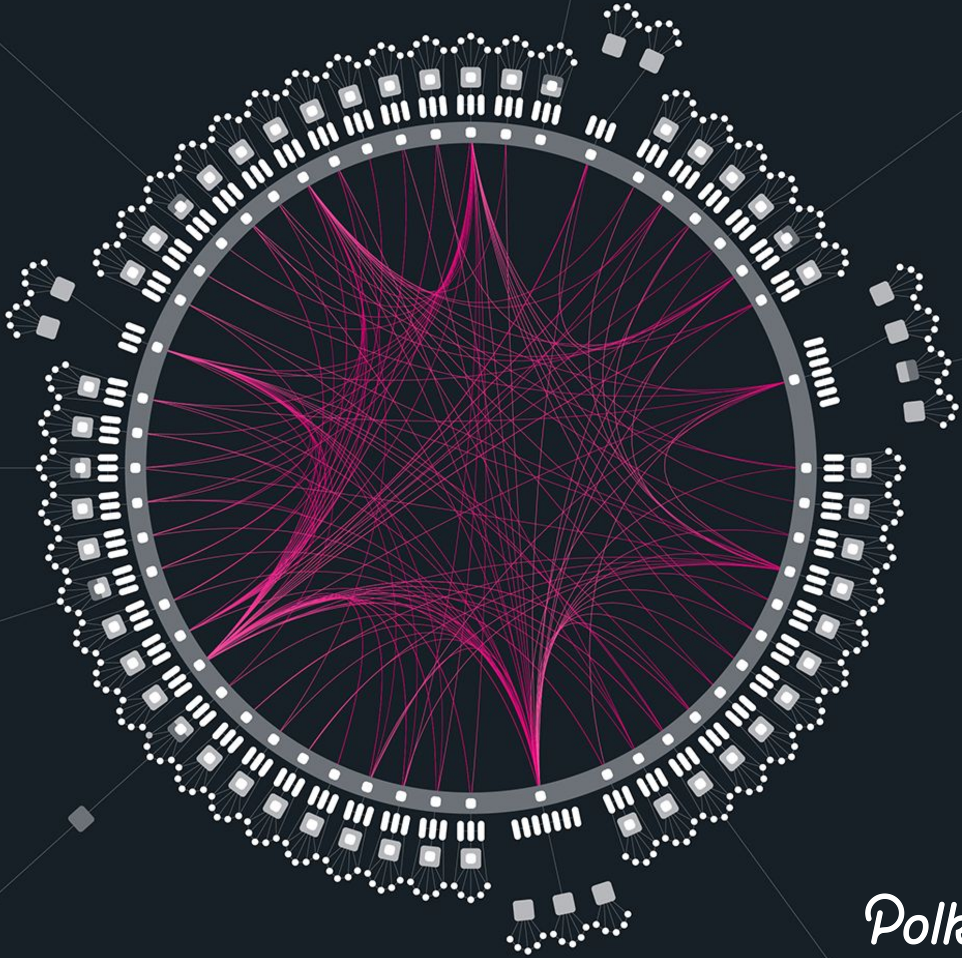
with an account balance
1.152 Mega

with a transaction nonce
0

submit the following extrinsic
system setCode(new) Set the new code.

new: Bytes
0x

SOME SUBSTRATE PROJECTS

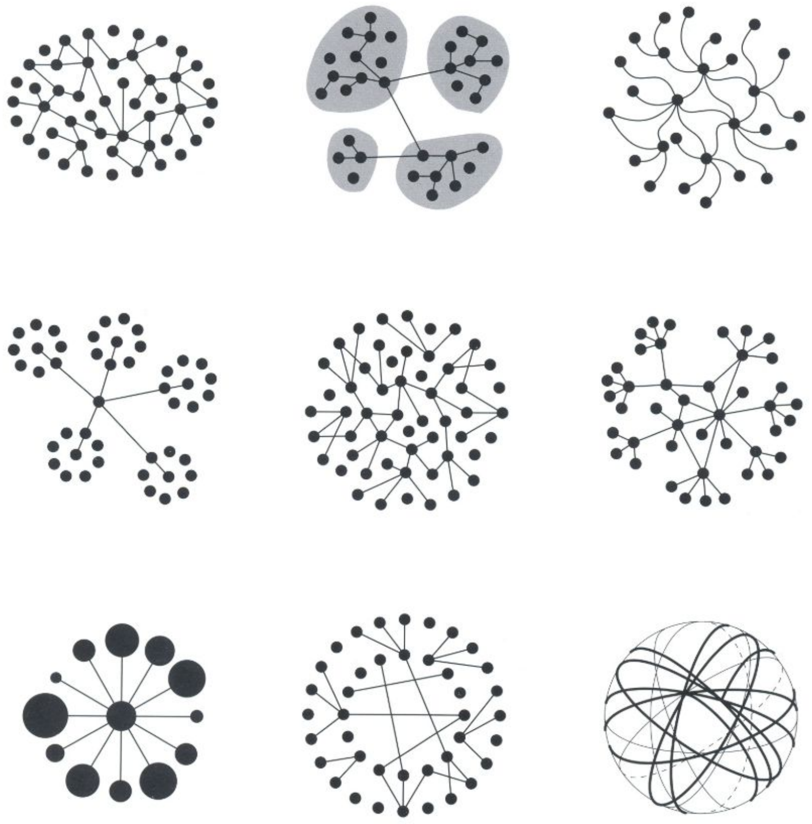


Polkadot.

Edgeware

A high-performance, self-upgrading WASM smart contract platform

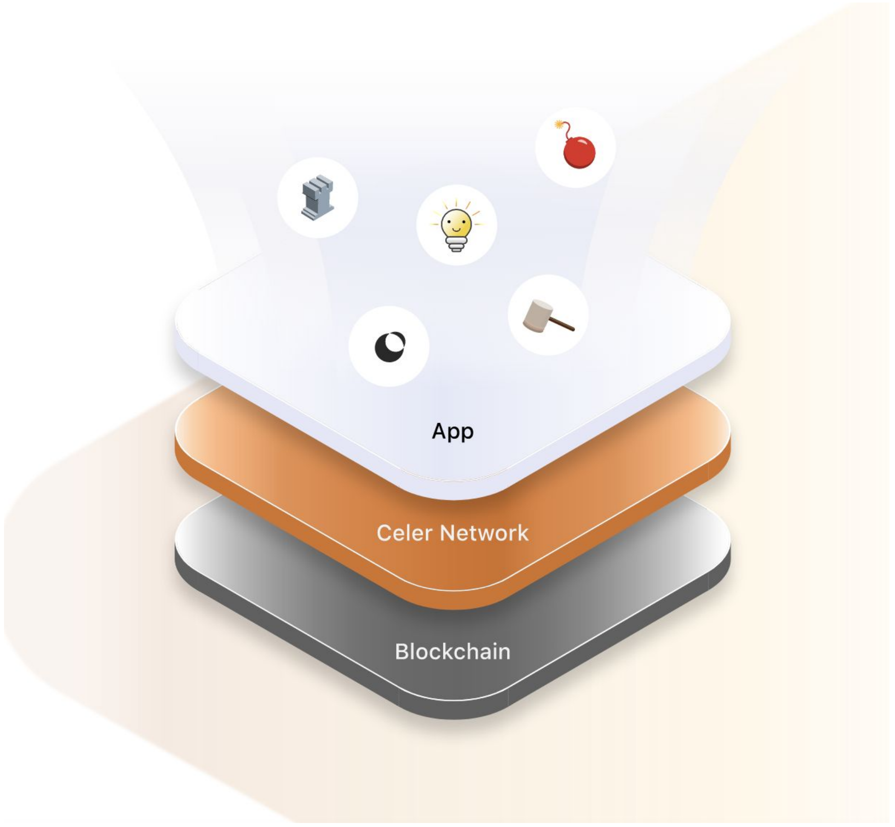
https://edgewa.re/



Celer Network

Layer-2 Scaling for Polkadot

https://www.celer.network/



Plasm

Plasma Scaling for Polkadot

From Stake Technologies

https://github.com/staketechnologies

Sunshine

Chain for fund coordination

DAOs from Web3 Garden

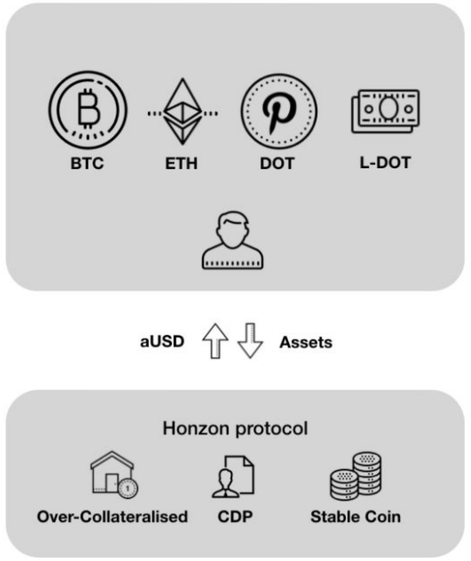
https://github.com/web3garden/sunshine-node

PLASM

Acala Network

Decentralized Stablecoin And Defi Network

https://acala.network/

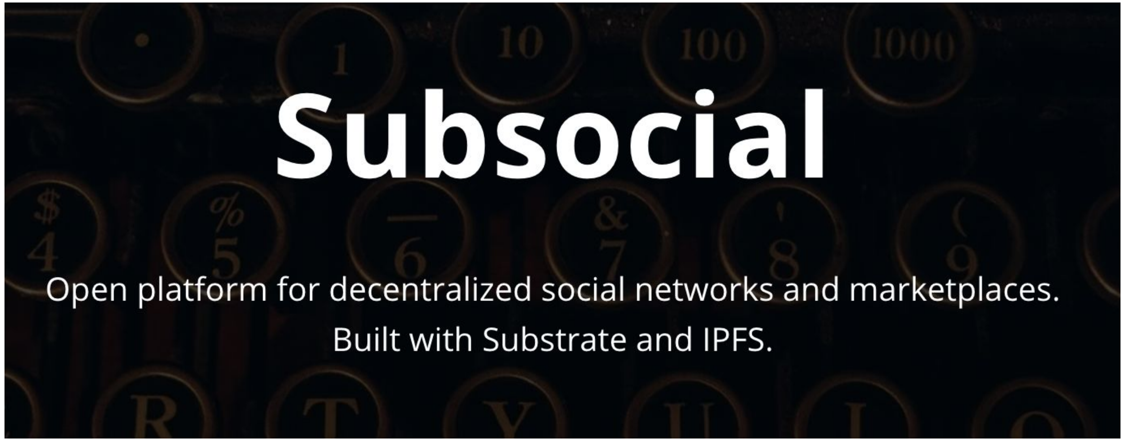


1 Acala Dollar = 1 US Dollar

Ticker : aUSD

Subsocial

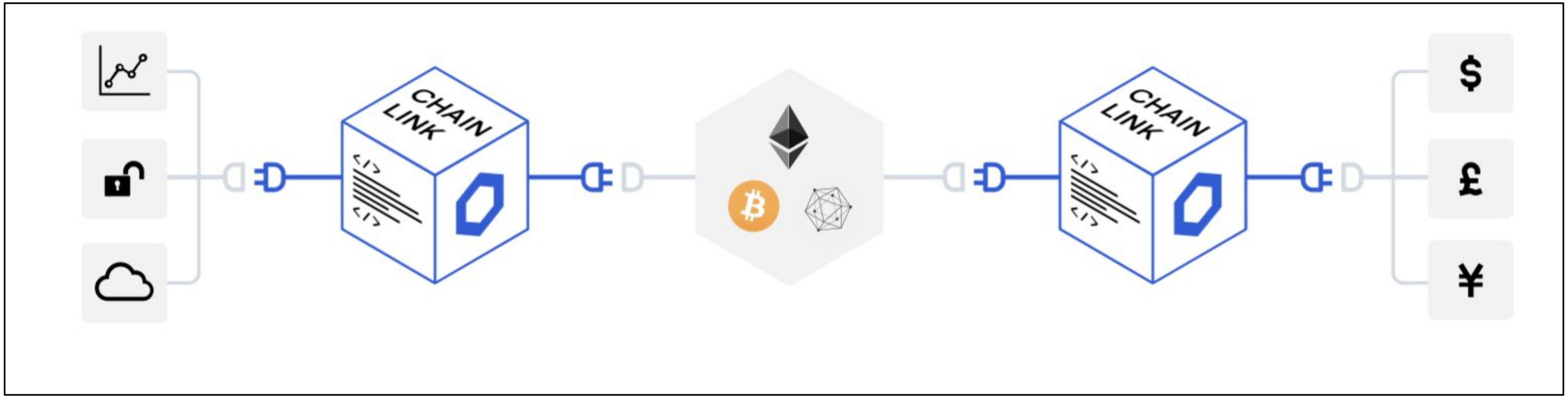
Allows anyone to launch a decentralized censorship-resistant community on their own blockchain.


Subsocial
Open platform for decentralized social networks and marketplaces.
Built with Substrate and IPFS.

http://subsocial.network/

ChainLink

Decentralized oracle service and payments processor



https://chain.link/

web3
foundation

GET IN TOUCH

Bill Laboon
Technical Education Lead at Web3 Foundation

Twitter: @BillLaboon
Email: bill@web3.foundation

Join us
https://polkadot.network
https://web3.foundation

Connect

An In-Depth Look At Substrate

web3
foundation

Web3 Foundation | https://web3.foundation | @web3foundation

Parity Technologies | https://parity.io | @paritytech

Why Rust?

Setting up Substrate

```
curl https://getsubstrate.io -sSf | bash -s -- --fast
```

```
WINDOWS: http://bit.ly/substrate-win
```

(~15-20 min)

Rust / Wasm

```
# Update Rust

rustup update nightly

rustup update stable

# Add Wasm target

rustup target add wasm32-unknown-unknown --toolchain nightly
```

Node template

```
cd substrate-node-template/

git checkout -b my-first-substrate-chain

cargo build --release
```

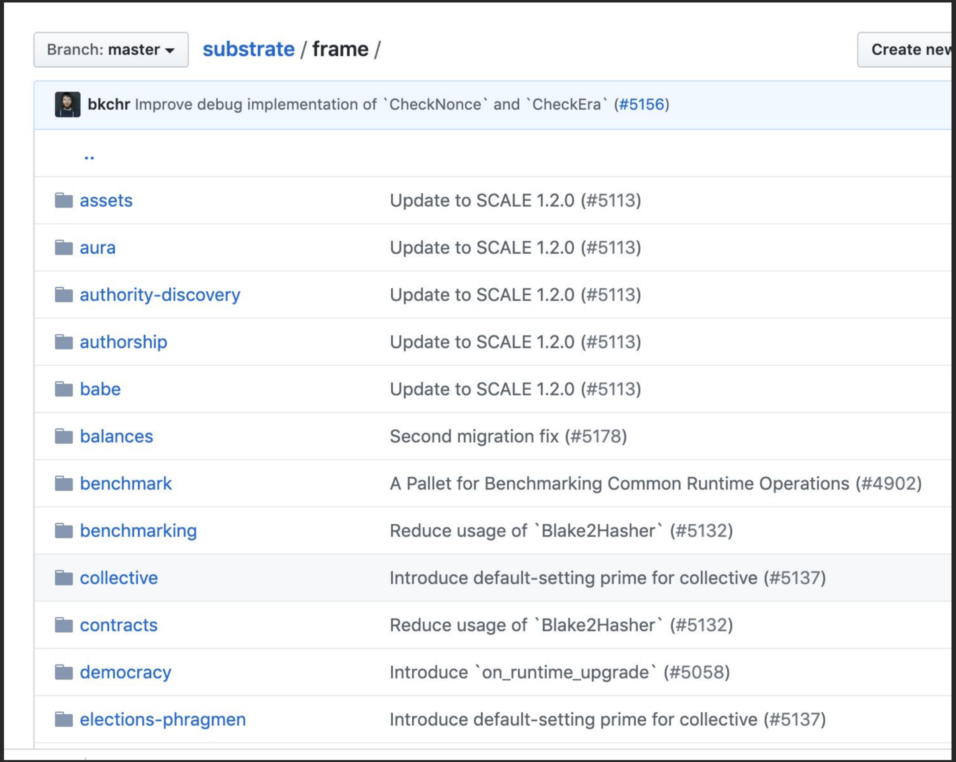
(~15-35 min)

Alternatively

`https://substrate-playground.w3f.community/`

(bookmark your URL after entry!)

Selecting Pallets



Branch: master ▾ substrate / frame / Create new

bkchr Improve debug implementation of `CheckNonce` and `CheckEra` (#5156)

..

assets	Update to SCALE 1.2.0 (#5113)
aura	Update to SCALE 1.2.0 (#5113)
authority-discovery	Update to SCALE 1.2.0 (#5113)
authorship	Update to SCALE 1.2.0 (#5113)
babe	Update to SCALE 1.2.0 (#5113)
balances	Second migration fix (#5178)
benchmark	A Pallet for Benchmarking Common Runtime Operations (#4902)
benchmarking	Reduce usage of `Blake2Hasher` (#5132)
collective	Introduce default-setting prime for collective (#5137)
contracts	Reduce usage of `Blake2Hasher` (#5132)
democracy	Introduce `on_runtime_upgrade` (#5058)
elections-phragmen	Introduce default-setting prime for collective (#5137)

Adding Pallets

```
Cargo.toml -

[features]
default = ["std"]
std = [
     "aura/std",
     "balances/std",
     "codec/std",
     "frame-executive/std",
     "frame-support/std",
     "grandpa/std",
     "randomness-collective-flip/std",
...
```

Building Your Own Pallet

```
// 1. Imports
use frame_support::{decl_module, decl_storage, decl_event,
dispatch::DispatchResult};
use system::ensure_signed;

// 2. Pallet Configuration
pub trait Trait: system::Trait { /* --snip-- */ }

// 3. Pallet Events
decl_event! { /* --snip-- */ }

// 4. Pallet Storage Items
decl_storage! { /* --snip-- */ }

// 5. Callable Pallet Functions
decl_module! { /* --snip-- */ }
```

Macros

`decl_storage!` `decl_module!` `decl_event!` `decl_error!`

- Rust code which can generate more code
- Used to simplify the creation of modules
- Can use custom syntax, not defined in Rust
- Hard to read the source code
- Treat them like magic

Rust is Explicit

You must tell Rust what to do in the case of any errors.

```rust
// Computes addition, returning the max value if overflow.
pub fn saturating_add(self, rhs: u8) -> u8;
// Computes addition, returning wrapped value if overflow.
pub fn wrapping_add(self, rhs: u8) -> u8;
// Computes addition, returning `None` if overflow.
pub fn checked_add(self, rhs: u8) -> Option<u8>;
```

Handling Errors in Your Runtime

- Your Runtime should never panic:
  - An unrecoverable error in Rust, which immediately terminates the thread
- Instead, you must perform “safe” operations which explicitly handles errors
- For example, safe math:

```
// BAD
let a = u8::max_value() + 1; // What should Rust do?
// GOOD
let a = u8::max_value().checked_add(1).ok_or("Overflow!")?;
```

Option Instead of Null

- Options let you be explicit about variables having some or no value

```
// Definition of Option type
enum Option<T> {
    Some(T),
    None,
}
```

```
let a = u8::max_value().checked_add(1)
a == None // True
let b = u8::max_value().checked_sub(1)
b == Some(254) // True
```

Result Instead of Panic

- Result is a richer version of Option that describes possible error instead of possible absence.

```
// Definition of Result
type
enum Result<T, E> {
    Ok(T),
    Err(E),
}
```

```
// Result in Substrate found in support::dispatch::Result
pub type Result = result::Result<(), &'static str>;
```

Verbose Error Handling

```
fn check_can_add(origin, x: u8, y: u8) -> Result {
    let a = match x.checked_add(y) {
        Some(v) => v,
        // Function caught an error, return Err(“message”)
        None => return Err("Overflow occurred")
    };
    // Function ran successfully, return Ok(())
    Ok(())
}
```

Simplified Result Handling

```rust
// These two expressions are equivalent
let sender = match ensure_signed(origin) {
    Ok(s) => s,
    Err(e) => return Err(e),
};

// Note the question mark (?) operator
let sender = ensure_signed(origin)?;
```

Basics of Runtime Development

Skeleton of a Module

```
use support::{decl_module, decl_storage, decl_event,...};
pub trait Trait: system::Trait {...}


decl_storage! {...}
decl_module! {...}
decl_event! {...}
decl_error! {...}
impl<T: Trait> Module<T> {...}
```

Declaring Storage

```
decl_storage! {
    trait Store for Module<T: Trait> as TemplateModule {
        // Here we are declaring a StorageValue, `SomeValue` as a u32
        // `get(some_value)` defines a getter function
        // Getter called with `Self::some_value()`
        SomeValue get(fn some_value): u32;
        // Here we are declaring a StorageMap from an AccountId to a Hash
        // Getter called with `Self::some_map(account_id)`
        SomeMap get(fn some_map): map T::AccountId => u32;
    }
}
```

Declaring Events

```
decl_event!(
    pub enum Event<T>
    where
        <T as system::Trait>::AccountId
    {
        // Event `ValueStored` deposits values of type `AccountId` and `u32`
        ValueStored(AccountId, u32),
    }
);
```

Declaring Dispatchable Functions

```
decl_module! {
  pub struct Module<T: Trait> for enum Call  where origin: T::Origin {
    fn deposit_event() = default; // The default deposit_event definition

    pub fn store_value(origin, input: u32) -> Result {
      let sender = ensure_signed(origin)?; // Check for transaction
      SomeValue::put(input); // Put a value into a StorageValue
      <SomeMap<T>>::insert(sender, input); // Insert key/value in StorageMap
      Self::deposit_event(RawEvent::ValueStored(sender, input)); // Emit
Event
      Ok(()) // Return Ok at the end of a function
}}}
```

Declaring Errors

```
decl_error! {
   /// Error for the identity module.
   pub enum Error for Module<T: Trait> {
       /// You can’t do that!
       YouCantDoThat,
       /// There was an overflow in the calculation.
       Overflow,
   }
}
```

Verify First, Write Last

- A “bad transaction” does not work the same as Ethereum
- Ethereum: State is reverted, storage is untouched, and a fee is paid
- Substrate: State changes will persist if an \`Err\` is returned
- Needed for situations like:
  - Increasing Account transaction nonce, even with failed transactions
  - Charging transaction fees even when “out of gas”
- Need to be conscious of this pattern when making “sub-functions”

Tutorials and More

Substrate.dev

- substrate.dev/recipes
- substrate.dev/seminar

Join Substrate's Riot Channel

web3
foundation

GET IN TOUCH

Bill Laboon
Technical Education Lead at Web3 Foundation

Twitter: @BillLaboon
Email: bill@web3.foundation

Join us
https://polkadot.network
https://web3.foundation

Connect