Solve · Cradle

CRADLESCRIPT · BLOCKCHAIN LIFE 2026 · DUBAI · 1–2 DECEMBER

Write a contract once. Ship it to every major chain.

EVM Solana NEAR Polkadot Aptos Sui

Every chain wants its own language. Every rewrite is a new bug.

EVM	Solana	NEAR	Polkadot	Aptos	Sui
Solidity	Rust · Anchor	near-sdk	ink!	Move	Move

Six codebases

The same token, vault or staking logic written again for each runtime, by different people.

Six audits

Each port is new code with its own reentrancy, access-control and arithmetic mistakes.

Six drifts

A fix on one chain has to be remembered, re-implemented and re-reviewed on the other five.

One chain-blind source.
Idiomatic output on each chain.

CRADLE compiles CradleScript, Rust or Solidity into one intermediate representation, checks it once, costs it per chain, then lowers it to native code for every target.

Write once

A focused language with its own keywords: blueprint, store, action, funded, ensure, dispatch.

Check once

Security analysis runs on the shared representation, so one rule protects every target.

Ship everywhere

Real PDAs and SPL CPI on Solana, real mappings on EVM, native Coin on Move. Not lowest common denominator.

Security sits on the build's critical path

Source

CradleScript · Rust · Solidity

Written once, with no chain in mind.

CRADLE IR

the keystone

One representation every adapter lowers out of.

USVE

23 analyzers

A high-severity finding blocks the build. Nothing unsafe gets lowered.

QRRO

cost report

Gas and compute units per chain, against real limits, before mainnet.

Adapters

EVM · SVM · WASM · Move

Idiomatic code for each of the six targets.

SIX TARGETS

Native on each chain

EVM
Solidity for Ethereum, BNB Chain, Polygon, Arbitrum, Optimism, Base and the rest.

Solana
A native program: PDAs, rent, SPL token CPI. Anchor-class.

NEAR
near-sdk, compiled to WASM.

Polkadot
ink!, compiled to WASM.

Aptos
Move, with native Coin handling.

Sui
Move, with native Coin handling.

A tip jar, written once

```
blueprint TipJar {
  store {
    owner: address protected = sender;
    collected: amount = 0;
  }
  funded action tip() {
    collected = checked_add(collected, received);
  }
  action withdraw(amount: amount) {
    ensure(sender == owner, NotOwner);
    ensure(collected >= amount, Insufficient);
    collected = checked_sub(collected, amount);
    dispatch(sender, amount);
  }
  read action total() -> amount { give collected; }
}
```

out/TipJar.sol

Payable function, checked transfer, owner guard.

out/tipjar_solana.rs

Lamports move from the program account to the owner.

tipjar 7/7

Every path run live on a validator and on an EVM node.

cradle build examples/tipjar.cradle

One command, both outputs.

PROVEN ON-CHAIN, NOT JUST COMPILED

Every capability has a conformance suite

Every suite runs in CI on every push, against a real Solana validator and an EVM node.

CRADLESCRIPT CAPABILITY	EVM	SOLANA	SUITE
State, roles, guards, records, time	Storage, modifiers	Account data, Borsh, Clock	`counter 7/7`
Per-key accounts	mapping(K ⇒ Struct)	One PDA per key	`pda_bank 7/7`
Native value in and out	Checked transfer	System-transfer CPI	`tipjar 7/7`
Fungible-token transfer	IERC20 transfer	SPL Token CPI	`token_vault 6/6`
Bounded loops	for loop	for loop	`batch 3/3`
All of the above, one program	—	—	`stake_pro 7/7`

BY THE NUMBERS

Small surface, wide reach

1

Source file per contract.

6

Target chain families: EVM, Solana, NEAR, Polkadot, Aptos, Sui.

23

Security checks on the IR before anything is lowered.

15

CI jobs green on every push, on-chain suites included.

One canonical address across every EVM chain

Same address everywhere

Deploy to Ethereum, BNB Chain, Polygon, Arbitrum or Base and get the same contract address on each.

Predict it before you deploy

Know the address ahead of time, so docs, frontends and partners can be ready on launch day.

Simpler SDKs and frontends

One address to configure, not a table of addresses per network.

Reproducible bytecode

Builds can be verified by anyone, byte for byte.

A web IDE that also writes the contract for you

No-code builder

Pick a template, fill a form. Studio writes an OpenZeppelin-based contract, scans it, deploys it and exports a dApp page.

AI builder

Describe the contract in plain English. It generates, compiles and fixes its own errors.

Chain chooser

EVM or Solana, switchable any time, with a clear pass or fail security gate.

Runs on your machine

Nothing leaves 127.0.0.1 until you deploy.

POWERED BY SOLIEUM

The native language of the Solieum Layer 2

The Solana program CRADLE emits is exactly what Solieum executes. Write once in CradleScript, run on Solieum, and ship the same source to every other major chain.

Solieum

Optimistic rollup that settles to Solana. Same SVM, so CRADLE output runs unchanged.

AI CONTROL

Its Trust Chain anchors AI decisions on Solieum. Contracts like these are how proofs get recorded.

Your dApp

Start in Studio, prove it on-chain, deploy to the chains your users are on.

Write it once. We'll show you the six outputs.

LANGUAGE solvelang.com

STUDIO cradle.solvelang.com

GUIDE cradle.solvelang.com/guide

TELEGRAM t.me/solieumlabs

MAIL support@solieum.com

Scan for the Solve brochure
solieum.com/brochures/solve

© 2026 SOLIEUM · SOLVE · CRADLE