

# On-chain vs. Off-chain

A conceptual overview of the roles and characteristics of on-chain and off-chain code in smart contracts.

*OpShin Pioneer Program*

*Source: Plutus Pioneer Program, PPP 040401*



# What do we mean by on-/off-chain?

## Roles of on-chain and off-chain code in a smart contract:

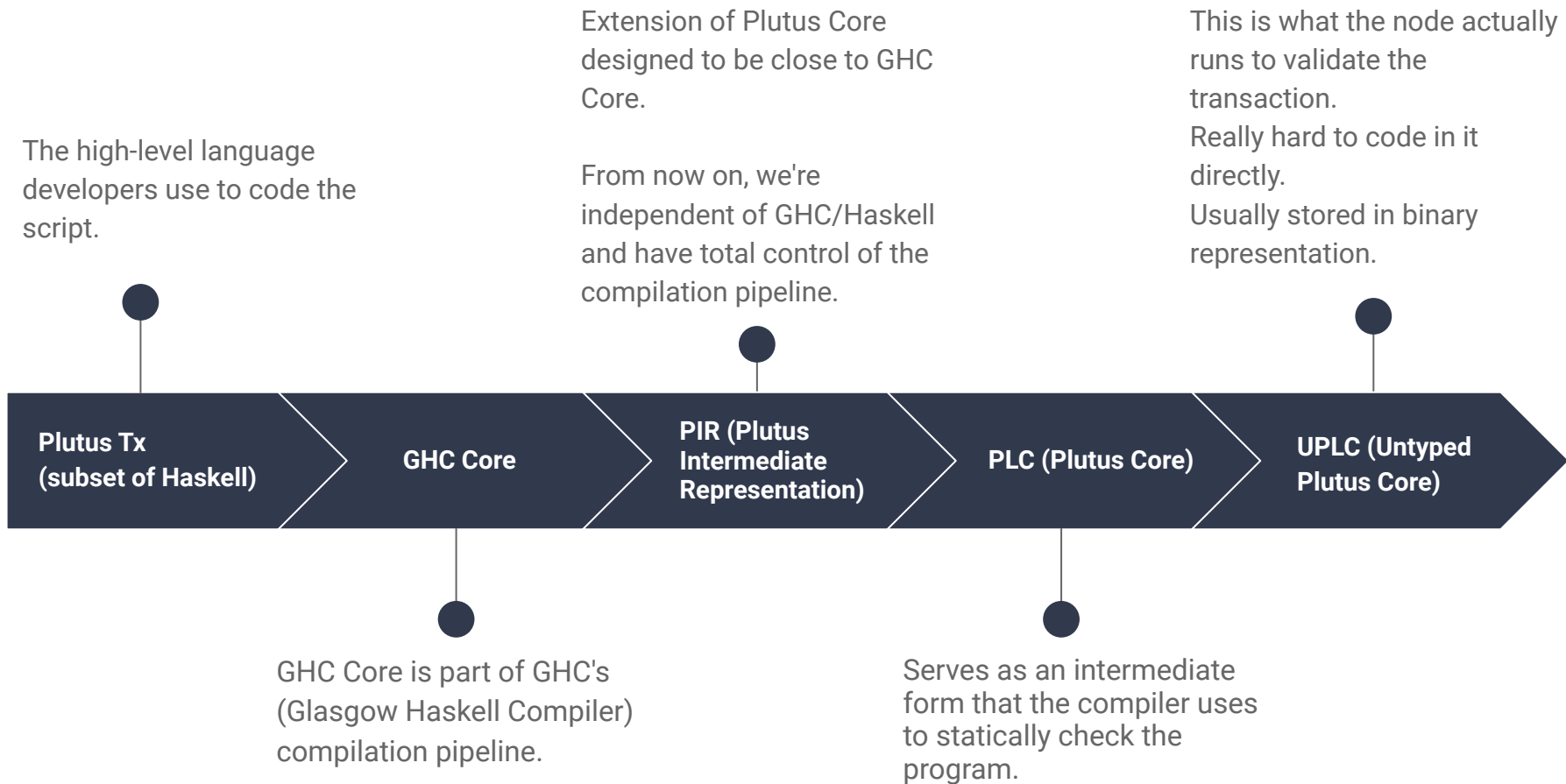
- **On-chain:** Code that runs in the node during the inclusion of new transactions and data that it's stored in the blockchain.
- **Off-chain:** Code that runs in the user's (or a service provider's) device to query the blockchain and build and submit transactions.

# Why is it necessary to decompose our code?

**Because both on-chain and off-chain code have inherent advantages and limitations.**

- **On-chain:** Ensures integrity, but it's expensive.
- **Off-chain:** Doesn't ensure integrity, but has access to the user's wallet and resources.

# Compiling On-Chain Code (the Plutus way)



# Compiling On-Chain Code (the OpShin way)

The high-level python-like language developers use to code the script.

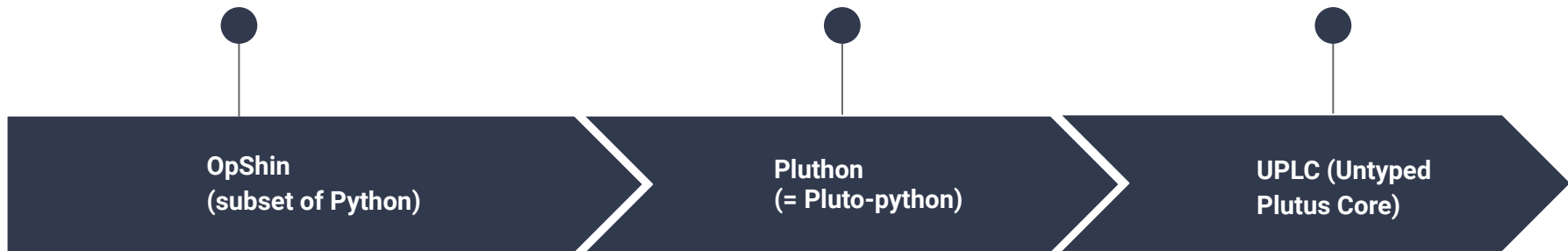
**OpShin**  
(subset of Python)

An intermediate language based on pluto (which is essentially a more human-readable way of writing UPLC directly).

**Pluthon**  
(= Pluto-python)

This is what the node actually runs to validate the transaction.  
Not human-friendly to code in it directly and usually stored in binary representation.

**UPLC (Untyped  
Plutus Core)**



cardano-cli query tip



```
class ChainContext:
    """Interfaces through which the library interacts with
    the Cardano blockchain."""

    def utxos(self, address: Union[str, Address]) -> List[UTx0]:
        """Get all UTxOs associated with an address."""
        return self._utxos(str(address))
```

## Off-chain: Querying the blockchain

### Getting everything we need:

- Cardano validators have access only to:
  - The context of the transaction we want to submit.
  - All its inputs and outputs (with their respective Values, Datums, and Scripts).
  - The Redeemer (our choice).

### There are many ways to get this information:

- Local node
- Local Database synchronized with a local or remote node using a chain indexer (e.g., db-sync, Kupo, Scrolls, Carp,...)
- Third-party tools that run their own nodes and databases (e.g., Blockfrost)

cardano-cli transaction build



```
# Build the transaction
builder = TransactionBuilder(context)
builder.add_script_input(
    utxo_to_spend,
    script=plutus_script,
    redeemer=redeemer
)
builder.collaterals.append(collateral_utxo)
builder.required_signers = [vkey_hash]
builder.validity_start = context.last_block_slot
builder.ttl = builder.validity_start + ttl_slots
```

## Off-chain: Building the Tx ("Use" validator)

- Indicating the inputs, reference inputs, and outputs
- Provide Datum
  - UTXO has datum's hash -> Provide the datum on Tx
  - UTXO has datum in Tx body -> Retrieve it from blockchain and provide it in Tx
  - UTXO has inline datum -> Indicate it's an inline datum (no need to provide it)
- Provide Script
  - No UTXO has the script attached -> Provide the actual script in Tx
  - We attached the script to a UTXO -> Provide the UTXO as a reference script
- Provide Redeemer
- Provide Collateral

cardano-cli transaction sign



```
signed_tx = builder.build_and_sign(  
    signing_keys=[payment_skey],  
    change_address=payment_address,  
)
```

## Off-chain: Signing the Tx

### Take into account:

- All transactions need to be signed by at least one key
- This is the only step that **MUST** be performed on the client (user's wallet) side
- The credentials should be stored in a cryptographic wallet (user's device)
- Provide a way to connect your off-chain code with wallets



cardano-cli transaction submit



```
# Submit the transaction  
context.submit_tx(signed_tx)
```

## Off-chain: Submitting the Tx

### Phase 1

- Checks whether the Tx was built correctly and can be added to the ledger.
- If it fails, Tx is rejected (no fees or collateral charged)

### Phase 2

- We have everything we need. Let's run the scripts!
- If it succeeds, the Tx is added to the ledger, and the fees are used to pay the node
- If it fails, the Tx is ignored, and collateral is used to pay the node

### Determinism

- Predictable effects -> Check phases off-chain