Writing Smart Contracts…
Without Solidity

macro

Agenda

- Why learn EVM opcodes?
- Contract create example
- Introducing: Trim
- Implementing the Greeter contract

Why learn EVM Opcodes?

To become a better Solidity engineer.

A Better Solidity Engineer

- Is better prepared for low-level hacks
- Has a deeper understanding of common design patterns
- Has internalized how smart contracts run on the EVM

Introducing: BASM

Instead of writing this…

```
0x600960405260206040f3
```

(This is EVM bytecode)

…you get to write this!

```
PUSH1 0x09
PUSH1 0x40
MSTORE
PUSH1 0x20
PUSH1 0x40
RETURN
```

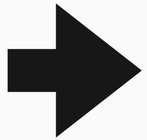
macro

BASM = Bare Assembly

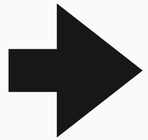
Introducing: BASM

…you get to write this!

```
PUSH1 0x09
PUSH1 0x40
MSTORE
PUSH1 0x20
PUSH1 0x40
RETURN
```



```
60 09
60 40
52
60 20
60 40
f3
```



```
0x6009604052602060 40f3
```

BASM = Bare Assembly

macro

Contract Create Example

```
PUSH1 0x0a
PUSH1 0x0c
PUSH1 0x00
CODECOPY
PUSH1 0x0a
PUSH1 0x00
RETURN
```

```
PUSH1 0x09
PUSH1 0x40
MSTORE
PUSH1 0x20
PUSH1 0x40
RETURN
```

Goal: Deploy a contract that always returns the number 9

Note that there are no “functions” in this contract.

```
Memory Usage
0x00: 0000000000000000000000000000000000
0x10: 0000000000000000000000000000000000
```

Contract Create Example

```
PUSH1 0x0a
PUSH1 0x0c
PUSH1 0x00
CODECOPY
PUSH1 0x0a
PUSH1 0x00
RETURN
```

Init bytecode

```
PUSH1 0x09
PUSH1 0x40
MSTORE
PUSH1 0x20
PUSH1 0x40
RETURN
```

Runtime bytecode

```
Memory Usage
0x00: 0000000000000000000000000000000000
0x10: 0000000000000000000000000000000000
```

macro

Goal: Deploy a contract that always returns the number 9

Contract Create Example

```
PUSH1 0x0a
PUSH1 0x0c
PUSH1 0x00
CODECOPY
PUSH1 0x0a
PUSH1 0x00
RETURN


PUSH1 0x09
PUSH1 0x40
MSTORE
PUSH1 0x20
PUSH1 0x40
RETURN
```

CODECOPY(destMemOffset: 0x00, codeOffset: 0x0c, codeLength: 0x0a)

```
Memory Usage
0x00: 0000000000000000000000000000000000
0x10: 0000000000000000000000000000000000
```

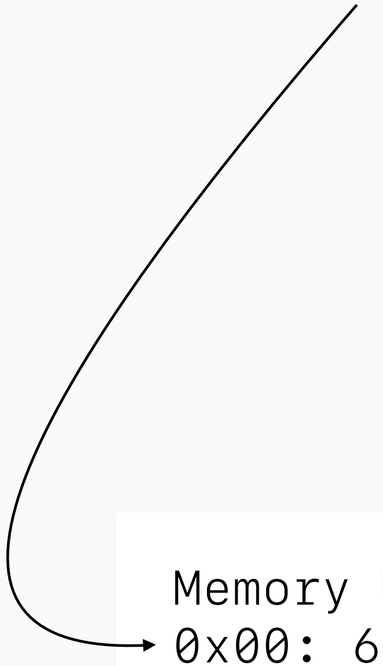
Goal: Deploy a contract that always returns the number 9

Contract Create Example

```
PUSH1 0x0a
PUSH1 0x0c
PUSH1 0x00
CODECOPY
PUSH1 0x0a
PUSH1 0x00
RETURN


PUSH1 0x09
PUSH1 0x40
MSTORE
PUSH1 0x20
PUSH1 0x40
RETURN
```

CODECOPY(destMemOffset: 0x00, codeOffset: 0x0c, codeLength: 0x0a)

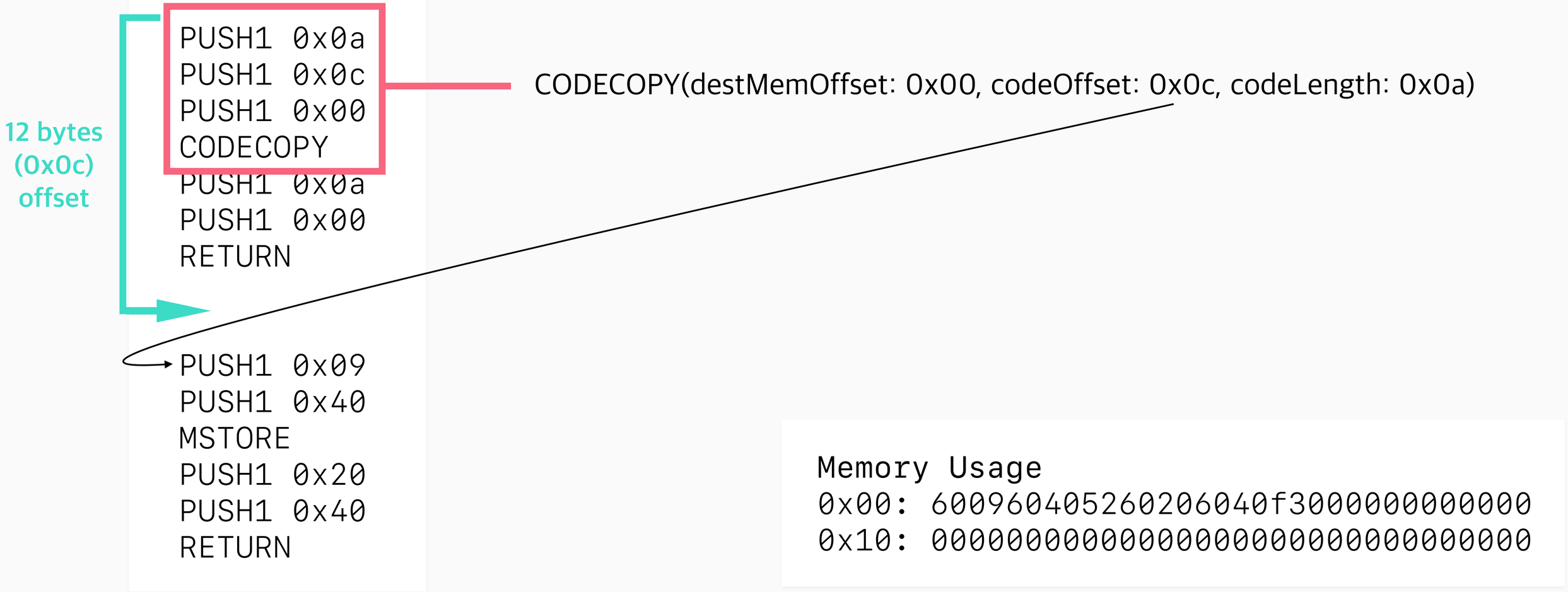


```
Memory Usage
0x00: 600960405260206040f3000000000000
0x10: 00000000000000000000000000000000
```

Goal: Deploy a contract that always returns the number 9

macro

Contract Create Example



Goal: Deploy a contract that always returns the number 9

macro

Contract Create Example

```
PUSH1 0x0a
PUSH1 0x0c
PUSH1 0x00
CODECOPY
PUSH1 0x0a
PUSH1 0x00
RETURN
```

CODECOPY(destMemOffset: 0x00, codeOffset: 0x0c, codeLength: 0x0a)

```
PUSH1 0x09
PUSH1 0x40
MSTORE
PUSH1 0x20
PUSH1 0x40
RETURN
```

10 bytes (0x0a) of code

```
Memory Usage
0x00: 6009604052602060400f3000000000000000
0x10: 0000000000000000000000000000000000
```

Goal: Deploy a contract that always returns the number 9

macro

Contract Create Example

```
PUSH1 0x0a
PUSH1 0x0c
PUSH1 0x00
CODECOPY
PUSH1 0x0a
PUSH1 0x00
RETURN
```

RETURN(memOffset: 0x00, length: 0x0a)

```
PUSH1 0x09
PUSH1 0x40
MSTORE
PUSH1 0x20
PUSH1 0x40
RETURN
```

```
Memory Usage
0x00: 6009604052602060 40f3000000000000
0x10: 0000000000000000 0000000000000000
```

Goal: Deploy a contract that always returns the number 9

macro

Contract Create Example

```
PUSH1 0x0a
PUSH1 0x0c
PUSH1 0x00
CODECOPY
PUSH1 0x0a
PUSH1 0x00
RETURN
```

RETURN(memOffset: 0x00, length: 0x0a)

```
PUSH1 0x09
PUSH1 0x40
MSTORE
PUSH1 0x20
PUSH1 0x40
RETURN
```

```
Memory Usage
0x00: 600960405260206040f3000000000000
0x10: 00000000000000000000000000000000
```

macro

Goal: Deploy a contract that always returns the number 9

Contract Create Example

```
PUSH1 0x0a
PUSH1 0x0c
PUSH1 0x00
CODECOPY
PUSH1 0x0a
PUSH1 0x00
RETURN


PUSH1 0x09
PUSH1 0x40
MSTORE
PUSH1 0x20
PUSH1 0x40
RETURN
```

Runtime bytecode length

Runtime bytecode offset

Runtime bytecode length

Problem: Manual byte counting!

```
Memory Usage
0x00: 600960405260206040f3000000000000
0x10: 00000000000000000000000000000000
```

Goal: Deploy a contract that always returns the number 9

Introducing: Trim

Instead of writing this…

```
PUSH1 0x0a
PUSH1 0x0c
PUSH1 0x00
CODECOPY
PUSH1 0x0a
PUSH1 0x00
RETURN


PUSH1 0x09
PUSH1 0x40
MSTORE
PUSH1 0x20
PUSH1 0x40
RETURN
```

…you get to write this!

```
(CODECOPY 0x00 0x0c 0x0a)
(RETURN 0x00 0x0a)


(MSTORE 0x40 0x09)
(RETURN 0x40 0x20)
```

macro

Feature 1: S-Expressions

Introducing: Trim

Instead of writing this…

```
(CODECOPY 0x00 0x0c 0x0a)
(RETURN 0x00 0x0a)


(MSTORE 0x40 0x09)
(RETURN 0x40 0x20)
```

…you get to write this!

```
(SUB CODESIZE #runtime)
DUP1
(CODECOPY 0x00 #runtime _)
(RETURN 0x00 _)


#runtime
(MSTORE 0x40 0x09)
(RETURN 0x40 0x20)
```

macro

Feature 2: Labels

Introducing: Trim

Instead of writing this…

```
; Push “Hello, Trim!”
PUSH12 0x48656c6c6f2c205472696d21
EQ
```

…you get to write this!

```
(EQ “Hello, Trim!” _)
```

Greeter Contract Example

```
contract Greeter {
  bytes32 private greeting = "Hello, EVM!";
  function greet() external view returns (bytes32) {
    return greeting;
  }
  function setGreeting(bytes32 _greeting) external {
    greeting = _greeting;
  }
}
```

Goal: Deploy a contract that implements this Solidity code behavior

Greeter Contract Example

Init storage slot zero

Runtime copying code

```
(SSTORE 0x00 “Hello, EVM!”)
(SUB CODESIZE #runtime)
DUP1
(CODECOPY 0x00 #runtime _)
(RETURN 0x00 _)

#runtime
; TODO
```

```
ABI Encoded function calls

greet() – 0xcfae3217

setGreeting(“Updated!”) –
0x50513b4f0000000000000000000000000000000000000000000000000000557064174656421
```

macro

Greeter Contract Example

Omit init code for slideshow

```
#runtime
; TODO
```

```
ABI Encoded function calls

greet() - 0xcfae3217

setGreeting(“Updated!”) -
0x50513b4f0000000000000000000000000000000000000000000000000000000557064174656421
```

Greeter Contract Example

```
#runtime
(CALLDATACOPY 0x1c 0x00 0x04)
(MLOAD 0x00)
```

Copy function selector onto stack

```
ABI Encoded function calls

greet() - 0xcfae3217

setGreeting("Updated!") -
0x50513b4f0000000000000000000000000000000000000000000000000000000000000000557064174656421
```

macro

Greeter Contract Example

```
#runtime
(CALLDATACOPY 0x1c 0x00 0x04)
(MLOAD 0x00)

(EQ 0xcfae3217 DUP1)
```

Check if it matches our known function selector

```
ABI Encoded function calls

greet() – 0xcfae3217

setGreeting(“Updated!”) –
0x50513b4f0000000000000000000000000000000000000000000000000000000000000000557064174656421
```

Greeter Contract Example

```
#runtime
(CALLDATACOPY 0x1c 0x00 0x04)
(MLOAD 0x00)

(EQ 0xcfae3217 DUP1)
(JUMPI #greet _)
```

If so, jump to the relevant code!

```
#greet
; TODO
#setGreeting
; TODO
```

```
ABI Encoded function calls

greet() – 0xcfae3217

setGreeting(“Updated!”) –
0x50513b4f0000000000000000000000000000000000000000000000000000000557064174656421
```

macro

Greeter Contract Example

```
#runtime
(CALLDATACOPY 0x1c 0x00 0x04)
(MLOAD 0x00)

(EQ 0xcfae3217 DUP1)
(JUMPI #greet _)

(EQ 0x50513b4f DUP1)
(JUMPI #setGreeting _)
```

Same idea for setGreeting

```
#greet
; TODO
#setGreeting
; TODO
```

```
ABI Encoded function calls

greet() – 0xcfae3217

setGreeting(“Updated!”) –
0x50513b4f0000000000000000000000000000000000000000000000000000000557064174656421
```

macro

Greeter Contract Example

```
#runtime
(CALLDATACOPY 0x1c 0x00 0x04)
(MLOAD 0x00)

(EQ 0xcfae3217 DUP1)
(JUMPI #greet _)

(EQ 0x50513b4f DUP1)
(JUMPI #setGreeting _)

REVERT  If no match, revert

#greet
; TODO
#setGreeting
; TODO
```

```
ABI Encoded function calls

greet() – 0xcfae3217

setGreeting(“Updated!”) –
0x50513b4f0000000000000000000000000000000000000000000000000000000557064174656421
```

macro

Greeter Contract Example

Omit for space

```
#greet
; TODO
#setGreeting
; TODO
```

```
ABI Encoded function calls

greet() – 0xcfae3217

setGreeting(“Updated!”) –
0x50513b4f0000000000000000000000000000000000000000000000000000000000000000557064174656421
```

macro

Greeter Contract Example

```
#greet
JUMPDEST
```

Required no-op

```
#setGreeting
; TODO
```

```
ABI Encoded function calls

greet() – 0xcfae3217

setGreeting(“Updated!”) –
0x50513b4f0000000000000000000000000000000000000000000000000000000000000557064617465421
```

macro

Greeter Contract Example

```
#greet
JUMPDEST
(MSTORE 0x20 (SLOAD 0x00))
```

Load current greeting into memory

```
#setGreeting
; TODO
```

```
ABI Encoded function calls

greet() – 0xcfae3217

setGreeting(“Updated!”) –
0x50513b4f0000000000000000000000000000000000000000000000000000000000000557064617465421
```

macro

Greeter Contract Example

```
#greet
JUMPDEST
(MSTORE 0x20 (SLOAD 0x00))
(RETURN 0x20 0x20)
```

Return the value!

```
#setGreeting
; TODO
```

```
ABI Encoded function calls

greet() – 0xcfae3217

setGreeting(“Updated!”) –
0x50513b4f0000000000000000000000000000000000000000000000000000000000000557064174656421
```

Greeter Contract Example

```
#greet
JUMPDEST
(MSTORE 0x20 (SLOAD 0x00))
(RETURN 0x20 0x20)


#setGreeting
JUMPDEST
```

Required no-op

```
ABI Encoded function calls

greet() – 0xcfae3217

setGreeting(“Updated!”) –
0x50513b4f0000000000000000000000000000000000000000000000000000000000000557064617465421
```

macro

Greeter Contract Example

```
#greet
JUMPDEST
(MSTORE 0x20 (SLOAD 0x00))
(RETURN 0x20 0x20)


#setGreeting
JUMPDEST
(SSTORE 0x0 (CALLDATALOAD 0x04))
```

Store ABI arg into storage slot zero

```
ABI Encoded function calls

greet() – 0xcfae3217

setGreeting(“Updated!”) –
0x50513b4f0000000000000000000000000000000000000000000000000557064174656421
```

macro

Greeter Contract Example

```
#greet
JUMPDEST
(MSTORE 0x20 (SLOAD 0x00))
(RETURN 0x20 0x20)


#setGreeting
JUMPDEST
(SSTORE 0x0 (CALLDATALOAD 0x04))
STOP
```

All done!

```
ABI Encoded function calls

greet() – 0xcfae3217

setGreeting(“Updated!”) –
0x50513b4f0000000000000000000000000000000000000000000000000000000000000557064617465421
```

macro

Greeter Contract Example

```
(SSTORE 0x00 "Hello, Trim!")
(SUB CODESIZE #runtime)
DUP1
(CODECOPY 0x00 #runtime _)
(RETURN 0x00 _)

#runtime
(CALLDATACOPY 0x1c 0x00 0x04)
(MLOAD 0x00) ; copy function id onto the stack

(EQ 0xcfae3217 DUP1)
(JUMPI #greet _)

DUP1
PUSH4 0x50513b4f ; function id for setGreeting
EQ
(JUMPI #setGreeting _)

REVERT ; If no matches, revert

#greet
JUMPDEST
(MSTORE 0x20 (SLOAD 0x00))
(RETURN 0x20 0x20)

#setGreeting
JUMPDEST
(SSTORE 0x00 (CALLDATALOAD 0x04))
STOP
```

So how did we do?

Solidity bytecode: 404 bytes

```
60806040527f48656c6c6f2c2045564d21000000000000000000000000000000000000000000600055
34801561003457600080fd5b506101508061004460003960 00f3fe6080604052348015610010576000
80fd5b506004361061003657600035 60e01c806350513b4f1461003b578063cfae321714610057575b
600080fd5b6100556004803603810190610050919061009d565b610075565b005b61005f61007f565b
60405161006c91906100d9565b60405180910390f35b8060008190555050565b60008054905090565b
600081359050610097816101035 65b92915050565b6000602082840312156100b3576100b26100fe56
5b5b60006100c184828501610088565b91505092915050565b6100d3816100f4565b82525050565b60
006020820190506100ee6000830184 6100ca565b92915050565b6000819050919050565b600080fd5b
61010c816100f4565b811461011757600080fd5b5056fea2646970667358221220 49247bb88def0472
669e897f88f4d397a2ac6835285c3561bac0a5c476cfa2cd64736f6c63430008070033
```

Trim bytecode: 85 bytes

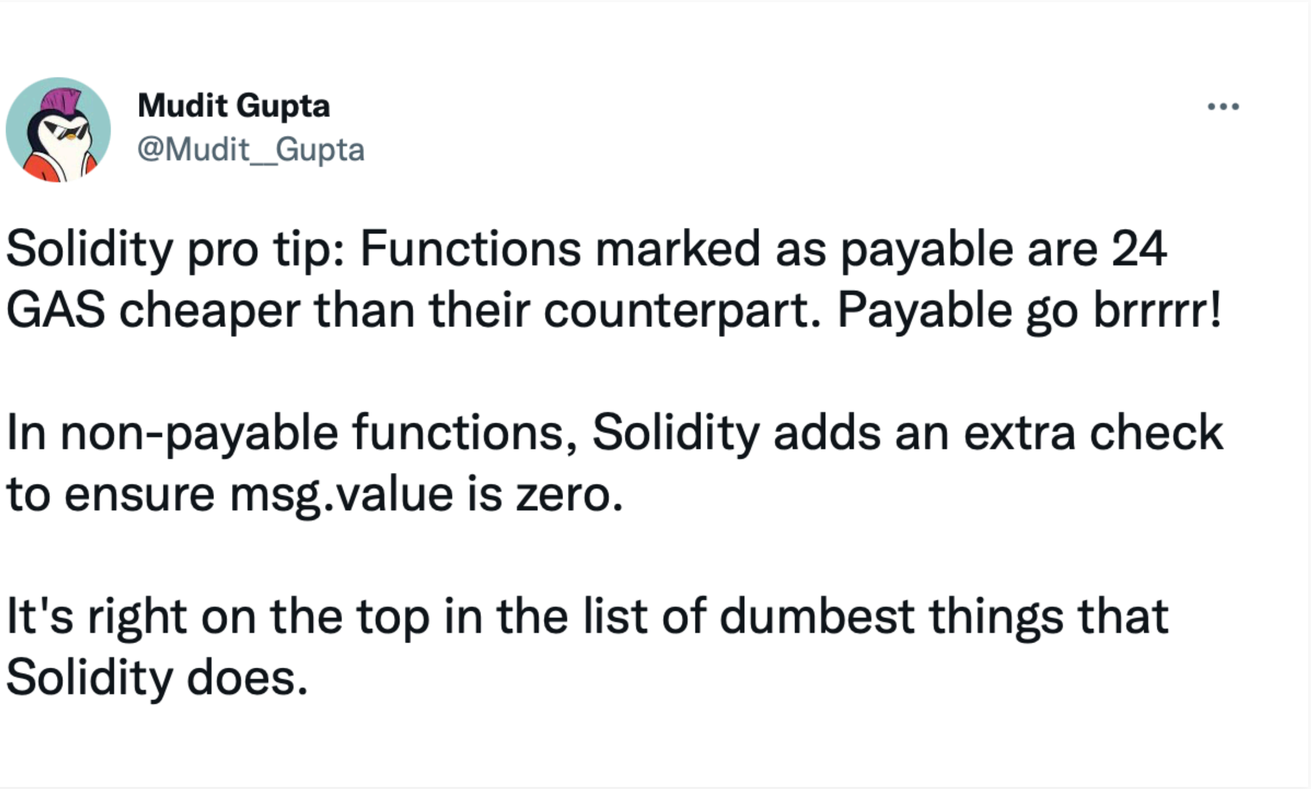
```
6b48656c6c6f2c205472696d2160005561001
f38038061001f6000396000f360046000601c
37600051806 3cfae32171461002157806 3505
13b4f1461002d57fd5b60005460205260206 0
20f35b60043560005500
```

However! Not the fairest comparison

Solidity generates
extra bytecode
for extra features

1. Non-payable checks
2. Calldata checks
3. Overflow/underflow checks

Etc. etc.



Conclusion

- Learning EVM opcodes helps you become a better Solidity engineer
- Trim is pretty cool
- Join the Smart Contract Engineering Fellowship at Macro!
  - *Find out more at 0xMacro.com!*

Thanks!

macro