Script:
Bitcoin's Programming Language

Ryan X. Charles
Blockchain University
Tokyo, Dec. 19, 2015

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Code Samples and Slides at:
github.com/ryanxcharles/blockchain-university

Datt

Outline

Script Interpreter
P2SH
Standard Transactions
Opcodes
Validating Transactions and Blocks
Advanced Scripts

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Datt

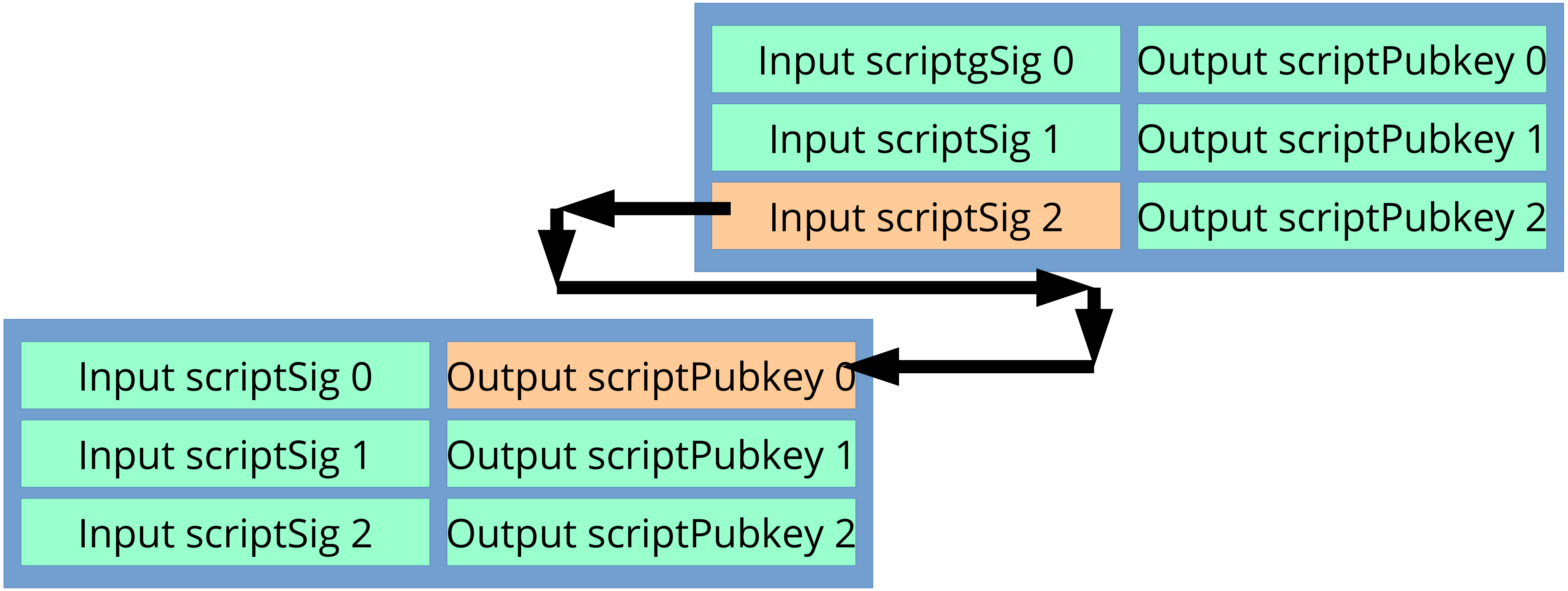
Script Interpreter

Input scriptSig 0	Output scriptPubkey 0
Input scriptSig 1	Output scriptPubkey 1
Input scriptSig 2	Output scriptPubkey 2

Transactions have inputs and outputs.
Inputs have scripts. Outputs have scripts.

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Script Interpreter



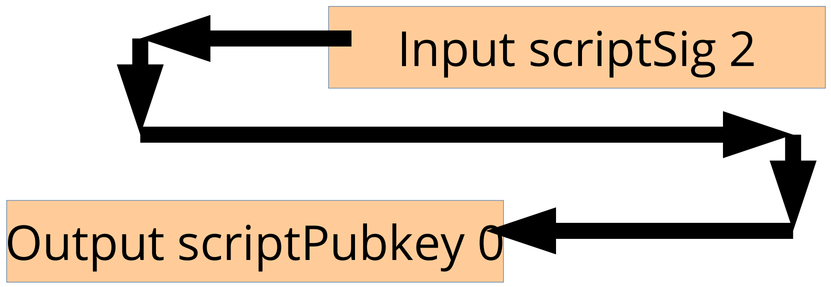
Each input links to the output of an earlier transaction.

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Code Samples and Slides at:
github.com/ryanxcharles/blockchain-university

Datt

Script Interpreter

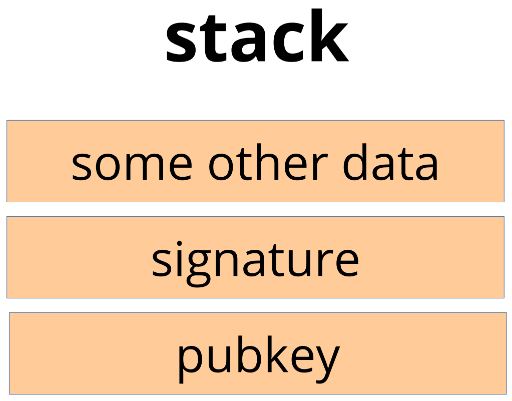


To validate an input, the scriptSig is executed and the scriptPubkey from the earlier transaction is executed.

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Script Interpreter

- The **stack** is the memory of bitcoin. Bitcoin does not have a heap. There is also an **alt stack** and things can be moved from stack to alt stack or from alt stack to stack.
- You can push and pop to the stack.
- Pubkeys and sigs are pushed to the stack. Other things like numbers can be pushed to the stack, if that's what the script does.



Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Datt

Script Interpreter

- To validate an input, the scriptSig is executed and the scriptPubkey from the earlier transaction is executed.
- After the scriptSig is executed, the stack is left the same, and the scriptPubkey runs starting with the same stack.
- Note that they are executed in “reverse” order – scriptSig first, from the later transaction, then scriptPubkey, from the earlier transaction

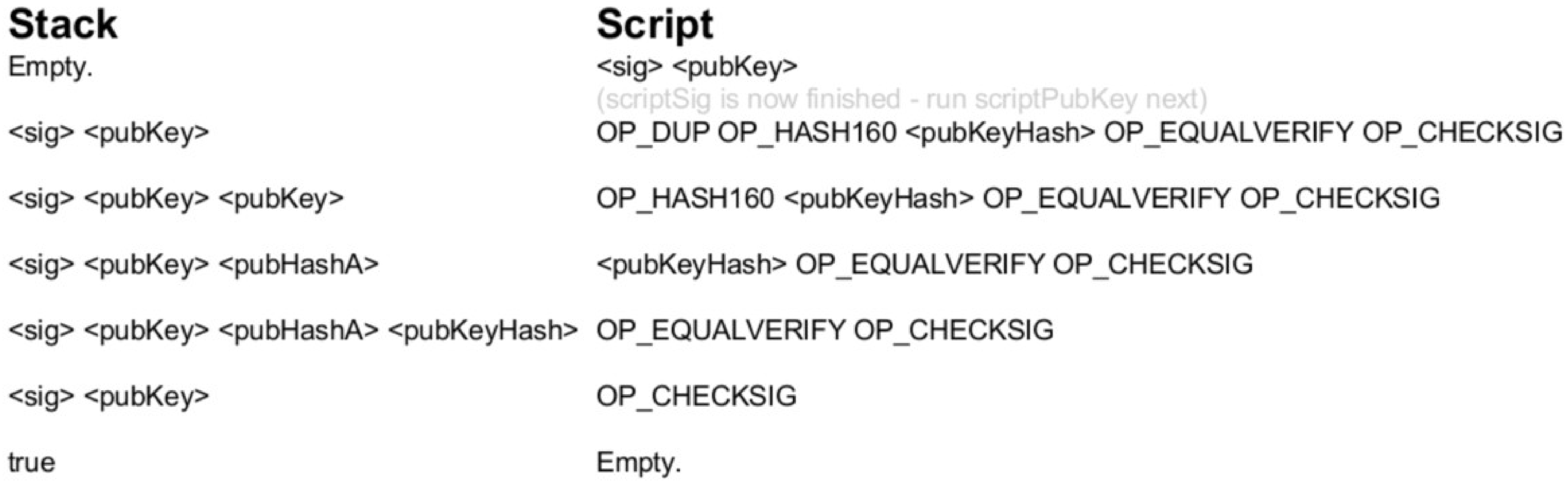
Input scriptSig 2	Output scriptPubkey 0

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Datt

Script Interpreter

pubkeyhash example, a.k.a. normal bitcoin address and transaction



Stack	Script
Empty.	<sig> <pubKey> (scriptSig is now finished - run scriptPubKey next)
<sig> <pubKey>	OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
<sig> <pubKey> <pubKey>	OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
<sig> <pubKey> <pubHashA>	<pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
<sig> <pubKey> <pubHashA> <pubKeyHash>	OP_EQUALVERIFY OP_CHECKSIG
<sig> <pubKey>	OP_CHECKSIG
true	Empty.

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Code Samples and Slides at:
github.com/ryanxcharles/blockchain-university

Datt

Opcodes

- One byte specifying an operation.

- OP_FALSE: 0x00,
- OP_0: 0x00,
- OP_PUSHDATA1: 0x4c,
- OP_PUSHDATA2: 0x4d,
- OP_PUSHDATA4: 0x4e,
- OP_1NEGATE: 0x4f,
- OP_RESERVED: 0x50,
- OP_TRUE: 0x51,
- OP_1: 0x51,
- OP_2: 0x52,
- OP_3: 0x53,
- OP_4: 0x54,
- OP_5: 0x55,
- OP_6: 0x56,
- OP_7: 0x57,
- OP_8: 0x58,
- OP_9: 0x59,
- OP_10: 0x5a,
- OP_11: 0x5b,
- OP_12: 0x5c,
- OP_13: 0x5d,
- OP_14: 0x5e,
- OP_15: 0x5f,
- OP_16: 0x60,
- OP_NOP: 0x61,
- OP_VER: 0x62,
- OP_IF: 0x63,
- OP_NOTIF: 0x64,
- OP_VERIF: 0x65,
- OP_VERNOTIF: 0x66,
- OP_ELSE: 0x67,
- OP_ENDIF: 0x68,
- OP_VERIFY: 0x69,
- OP_RETURN: 0x6a,
- ... ~175 total, including push data

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Datt

Push Ops

- 0 – 0x80: That many bytes are pushed to the stack.
  - e.g., 0x05 0x0404040404 pushes “0404040404” to the stack
- PUSHDATA1: The following byte specifies amount of data to push
  - e.g.: OP_PUSHDATA1 0x03 0x010203 pushes “010203” to the stack
- PUSHDATA2: The following two bytes (Uint16BE) specify the amount of data to push
- PUSHDATA4: The following four bytes (Uint32BE) specify the amount of data to push
- Anything 80 bytes or less can be pushed with a single byte push OP rather than PUSHDATAX – that includes signatures, usually ~70 bytes, and pubkeys, ~33 or ~65 bytes. p2sh redeemScripts, often containing multiple pubkeys, require PUSHDATA1 or PUSHDATA2

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Datt

Number Ops

- 0 = 0x00 = OP_0 = OP_FALSE
- OP_1 = 81 = 0x51 = OP_TRUE
- OP_2 = 82 = 0x52
- …
- OP_16 = 96 = 0x60
- Pushes that number to the stack. The number is a ScriptNum – zero bytes if 0, one byte if 1 - 16

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Control Ops

- OP_IF: 0x63 – checks that top item on stack is true
- OP_NOTIF: 0x64
- OP_ELSE: 0x67
- OP_ENDIF: 0x68
- OP_RETURN: 0x6a ← **commonly used**

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Stack Ops

- Rearrange things on the stack or alt stack
- OP_TOALTSTACK: 0x6b
- OP_FROMALTSTACK: 0x6c
- OP_2DROP: 0x6d
- OP_2DUP: 0x6e
- OP_3DUP: 0x6f
- OP_2OVER: 0x70
- OP_2ROT: 0x71
- OP_2SWAP: 0x72
- OP_IFDUP: 0x73
- OP_DEPTH: 0x74
- OP_DROP: 0x75
- OP_DUP: 0x76 ← **commonly used**
- OP_NIP: 0x77
- OP_OVER: 0x78
- OP_PICK: 0x79
- OP_ROLL: 0x7a
- OP_ROT: 0x7b
- OP_SWAP: 0x7c
- OP_TUCK: 0x7d

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Code Samples and Slides at:
github.com/ryanxcharles/blockchain-university

Datt

Splice Ops

- Rearrange bytes of the top stack item
- OP_CAT: 0x7e
- OP_SUBSTR: 0x7f
- OP_LEFT: 0x80
- OP_RIGHT: 0x81
- OP_SIZE: 0x82

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Datt

Bitwise Ops

- Rearrange bytes of the top stack item
- OP_INVERT: 0x83
- OP_AND: 0x84
- OP_OR: 0x85
- OP_XOR: 0x86
- OP_EQUAL: 0x87
- OP_EQUALVERIFY: 0x88, ← **commonly used**
- OP_RESERVED1: 0x89
- OP_RESERVED2: 0x8a

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Datt

Numeric Ops

- Not commonly used – some are disabled

- OP_1ADD: 0x8b
- OP_1SUB: 0x8c
- OP_2MUL: 0x8d
- OP_2DIV: 0x8e
- OP_NEGATE: 0x8f
- OP_ABS: 0x90
- OP_NOT: 0x91
- OP_0NOTEQUAL: 0x92

- OP_ADD: 0x93
- OP_SUB: 0x94
- OP_MUL: 0x95
- OP_DIV: 0x96
- OP_MOD: 0x97
- OP_LSHIFT: 0x98
- OP_RSHIFT: 0x99

- OP_BOOLAND: 0x9a
- OP_BOOLOR: 0x9b
- OP_NUMEQUAL: 0x9c
- OP_NUMEQUALVERIFY: 0x9d
- OP_NUMNOTEQUAL: 0x9e
- OP_LESSTHAN: 0x9f
- OP_GREATERTHAN: 0xa0
- OP_LESSTHANOREQUAL: 0xa1
- OP_GREATERTHANOREQUAL: 0xa2
- OP_MIN: 0xa3
- OP_MAX: 0xa4

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Cryptography Ops

- OP_RIPEMD160: 0xa6
- OP_SHA1: 0xa7
- OP_SHA256: 0xa8
- OP_HASH160: 0xa9 ← **commonly used**
- OP_HASH256: 0xaa
- OP_CODESEPARATOR: 0xab
- OP_CHECKSIG: 0xac ← **commonly used**
- OP_CHECKSIGVERIFY: 0xad
- OP_CHECKMULTISIG: 0xae ← **commonly used**
- OP_CHECKMULTISIGVERIFY: 0xaf

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Standard Script Types

- pubkeyhash ← most common
- pubkey
- multisig
- p2sh
- OP_RETURN

- Since p2sh redeemScript is itself a script, combinations like “p2sh multisig” are possible – p2sh multisig is most common use of p2sh

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Datt

pubkeyhash

- scriptSig:
- <sig> <pubkey>

- scriptPubkey:
- OP_DUP OP_HASH160 <address> OP_EQUALVERIFY OP_CHECKSIG

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Datt

pubkey (rare)

- scriptSig:
- <sig>

- scriptPubkey:
- <pubkey> OP_CHECKSIG

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Code Samples and Slides at:
github.com/ryanxcharles/blockchain-university

Datt

multisig

- scriptSig:
- OP_0 <sig1> <sig2> ... <sigm> ← starts with extra OP_0 because of famous multisig bug – pops one too many items from stack

- scriptPubkey:
- OP_m <pubkey1> <pubkey2> ... <pubkeyn> OP_n OP_CHECKMULTISIG

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Datt

p2sh multisig

- scriptSig:
- OP_0 <sig1> <sig2> ... <sigm> <redeemScript>

- scriptPubkey:
- OP_HASH160 <redeemScriptHash> OP_EQUALVERIFY

- redeemScript:
- OP_m <pubkey1> <pubkey2> ... <pubkeyn> OP_n OP_CHECKMULTISIG

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

OP_RETURN

- scriptPubkey:
- OP_RETURN <up to 40 (80?) bytes of data>

- How to put arbitrary data in an output if necessary.

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Datt

Standard Transaction Rules

- If a transaction has all standard inputs/outputs, it is standard
- Standard transactions are relayed by default
- Non-standard transactions can still be valid, and can be in a block, if a miner receives it somehow and chooses to include it
- Complicated scripts are thus discouraged; preventing hypothetical DOS attacks on the network and blockchain bloat

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Datt

Validating a Transaction

- Be sure that there is at least one input
- Be sure that transaction is not over MAX_BLOCK_SIZE
- Be sure that values are not negative or greater than MAX_MONEY
- Be sure inputs are not duplicated
- Be sure that inputs are not null
- Run script interpreter on all inputs and be sure no inputs are invalid

Ryan X. Charles
Founder of Datt (datt.co)
twitter.com/ryanxcharles
github.com/ryanxcharles

Datt