Bitcoin Script

Bitcoin uses it's own stack-based scripting language with a set of op-codes. This scripting language is used to determine whether an unspent transaction output (utxo) can be spent. When you send Bitcoin, you are really just assigning a Bitcoin value to a script, which can be spent only if the script executes successfully and results in a single true value on the stack. The script that is assigned to the Bitcoin transaction output is called the `scriptPubKey`. To satisfy the script, you must provide the correct `scriptSig`, which is also a script, which is prepended to the `scriptSig` and executed to test if the spending succeeded. Let's look at an example:

`scriptPubKey (raw): 76a914306e2ea1eed91bf66dfe5d94f3957d4ba63bde8488ac scriptPubKey (assembly): OP_DUP OP_HASH160 306e2ea1eed91bf66dfe5d94f3957d4ba63bde84 OP_EQUALVERIFY OP_CHECKSIG`

In plain english, this script, when executed will do the following:

- OP_DUP: Duplicate the top item on the stack (place a copy of it on the stack)
- OP_HASH160: Pop the top item off the stack, calculate it's HASH160, i.e. the RIPEMD160(SHA256()), and place that on the stack
- OP_EQUALVERIFY: Pop the top 2 items off the stack, and if they are not equal, it will exit immediately and fail, otherwise nothing
- OP_CHECKSIG: Pop the top 2 items off the stack, and use the top as the public key, the second as the signature, and verify the transaction signature

In order to spend this output, we must provide a valid signature and public key. We'll just provide `00` for the signature for this example, and not worry about the final checksig op.

Let's visualize this example with a `scriptSig`. Let's use kallewoof's `btcdeb` Script visualization tool. After the first command, you can type `step` once, and then just press enter each time after that to step through each operation:

```
$ btcdeb '[00 030cfcefa07af9dd6dbe770b87d7dbdd2c31ba7f4fcf8f3a1196d502f13561b046 OP_DUP OP_HASH160
306e2ea1eed91bf66dfe5d94f3957d4ba63bde84 OP_EQUALVERIFY OP_CHECKSIG]'

btcdeb -- type `btcdeb -h` for start up options
valid script
7 op script loaded. type `help` for usage information
script                                                                |  stack
----------------------------------------------------------------------+--------
0                                                                     |
030cfcefa07af9dd6dbe770b87d7dbdd2c31ba7f4fcf8f3a1196d502f13561b046 |
OP_DUP                                                                |
OP_HASH160                                                            |
306e2ea1eed91bf66dfe5d94f3957d4ba63bde84                              |
OP_EQUALVERIFY                                                        |
OP_CHECKSIG                                                           |
#0000 0
btcdeb> step
                <> PUSH stack
script                                                                |  stack
----------------------------------------------------------------------+--------
030cfcefa07af9dd6dbe770b87d7dbdd2c31ba7f4fcf8f3a1196d502f13561b046 |      0x
OP_DUP                                                                |
OP_HASH160                                                            |
306e2ea1eed91bf66dfe5d94f3957d4ba63bde84                              |
OP_EQUALVERIFY                                                        |
OP_CHECKSIG                                                           |
#0001 030cfcefa07af9dd6dbe770b87d7dbdd2c31ba7f4fcf8f3a1196d502f13561b046
```

```
btcdeb>
                <> PUSH stack 030cfcefa07af9dd6dbe770b87d7dbdd2c31ba7f4fcf8f3a1196d502f13561b046
script                                                                |
stack
----------------------------------------------------------------------+----------------------------------
--------------------------------
OP_DUP                                                                |
030cfcefa07af9dd6dbe770b87d7dbdd2c31ba7f4fcf8f3a1196d502f13561b046
OP_HASH160                                                            |
0x
306e2ea1eed91bf66dfe5d94f3957d4ba63bde84                              |
OP_EQUALVERIFY                                                        |
OP_CHECKSIG                                                           |
#0002 OP_DUP
btcdeb>
                <> PUSH stack 030cfcefa07af9dd6dbe770b87d7dbdd2c31ba7f4fcf8f3a1196d502f13561b046
script                                                                |
stack
----------------------------------------------------------------------+----------------------------------
--------------------------------
OP_HASH160                                                            |
030cfcefa07af9dd6dbe770b87d7dbdd2c31ba7f4fcf8f3a1196d502f13561b046
306e2ea1eed91bf66dfe5d94f3957d4ba63bde84                              |
030cfcefa07af9dd6dbe770b87d7dbdd2c31ba7f4fcf8f3a1196d502f13561b046
OP_EQUALVERIFY                                                        |
0x
OP_CHECKSIG                                                           |
#0003 OP_HASH160
btcdeb>
                <> POP  stack
                <> PUSH stack 306e2ea1eed91bf66dfe5d94f3957d4ba63bde84
script                                                                |
stack
----------------------------------------------------------------------+----------------------------------
--------------------------------
306e2ea1eed91bf66dfe5d94f3957d4ba63bde84                              |
306e2ea1eed91bf66dfe5d94f3957d4ba63bde84
OP_EQUALVERIFY                                                        |
030cfcefa07af9dd6dbe770b87d7dbdd2c31ba7f4fcf8f3a1196d502f13561b046
OP_CHECKSIG                                                           |
0x
#0004 306e2ea1eed91bf66dfe5d94f3957d4ba63bde84
btcdeb>
                <> PUSH stack 306e2ea1eed91bf66dfe5d94f3957d4ba63bde84
script                                                                |
stack
----------------------------------------------------------------------+----------------------------------
--------------------------------
OP_EQUALVERIFY                                                        |
306e2ea1eed91bf66dfe5d94f3957d4ba63bde84
OP_CHECKSIG                                                           |
306e2ea1eed91bf66dfe5d94f3957d4ba63bde84
                                                                      |
030cfcefa07af9dd6dbe770b87d7dbdd2c31ba7f4fcf8f3a1196d502f13561b046
                                                                      |
0x
#0005 OP_EQUALVERIFY
btcdeb>
                <> POP  stack
                <> POP  stack
                <> PUSH stack 01
                <> POP  stack
script                                                                |
stack
----------------------------------------------------------------------+----------------------------------
--------------------------------
OP_CHECKSIG                                                           |
030cfcefa07af9dd6dbe770b87d7dbdd2c31ba7f4fcf8f3a1196d502f13561b046
                                                                      |
```

```
0x
#0006 OP_CHECKSIG
```

You can play around with this using any of the opcodes listed at Script.

TODO: Create an actual transaction and provide the raw hex to do the checksig. Also show different ways the transaction can fail.