marp
true

Calldata (how to target functions!)

- its helpful to remember that Solidity's job is to compile contracts to bytecode
- Solidity doesnt know about the chain its deployed on
- if you tell solidity to call an address with calldata, it will do that
- you do this with both the high-level and low-level syntax

High Level Syntax

First off, the high level syntax for message calls targetting functions:

```
contract A {
  uint sum;
  function storeSum(address b) external {
    sum = B(b).add(5, 10);
  }
}

contract B {
  function add(uint x, uint y) external returns(uint) {
    return x + y;
  }
}
```

Same thing

The argument is an address either way:

```
function storeSum(B b) external {
  sum = b.add(5, 10);
}
```

Or:

```
function storeSum(address b) external {
  sum = B(b).add(5, 10);
}
```

Also, interfaces

The argument is an address either way:

```
contract A {
  uint sum;
  function storeSum(B b) external {
    sum = b.add(5, 10);
  }
}

interface B {
  function add(uint, uint) external returns(uint);
}
```

👆 this is a message call, you are defining calldata

Low Level Syntax

Ok, so how about the low-level way?

```
contract A {
  uint sum;
  function storeSum(address b) external {
    (bool success, bytes memory returnData) =
      b.call(abi.encodeWithSignature("add(uint256,uint256)", 5, 10));
```

```
      sum = abi.decode(returnData, (uint));
      require(success);
    }
  }

  contract B {
    function add(uint x, uint y) external pure returns(uint) {
      return x + y;
    }
  }
```

EncodeWithSignature Breakdown

What is `abi.encodeWithSignature` doing? It is combining:

- first 4 bytes of the `keccak256` of the `add` method - `0x771602f7`
- the first argument as a uint256 - `0x0000000000000000000000000000000000000000000000000000000000000005`
- the second argument as a uint256 - `0x000000000000000000000000000000000000000000000000000000000000000a`

Final calldata:
`0x771602f70000000000000000000000000000000000000000000000000000000000000005000000000000000000000000000000000000000000000000000000000000000a`

Sending Calldata

Regardless of which syntax you use, solidity is compiling a contract to send some calldata `0x771602f70000000000000000000000000000000000000000000000000000000000000005000000000000000000000000000000000000000000000000000000000000000a` to some address.

🧑‍💻 It's up to you, as a developer, to make sure that contract responds to that calldata.