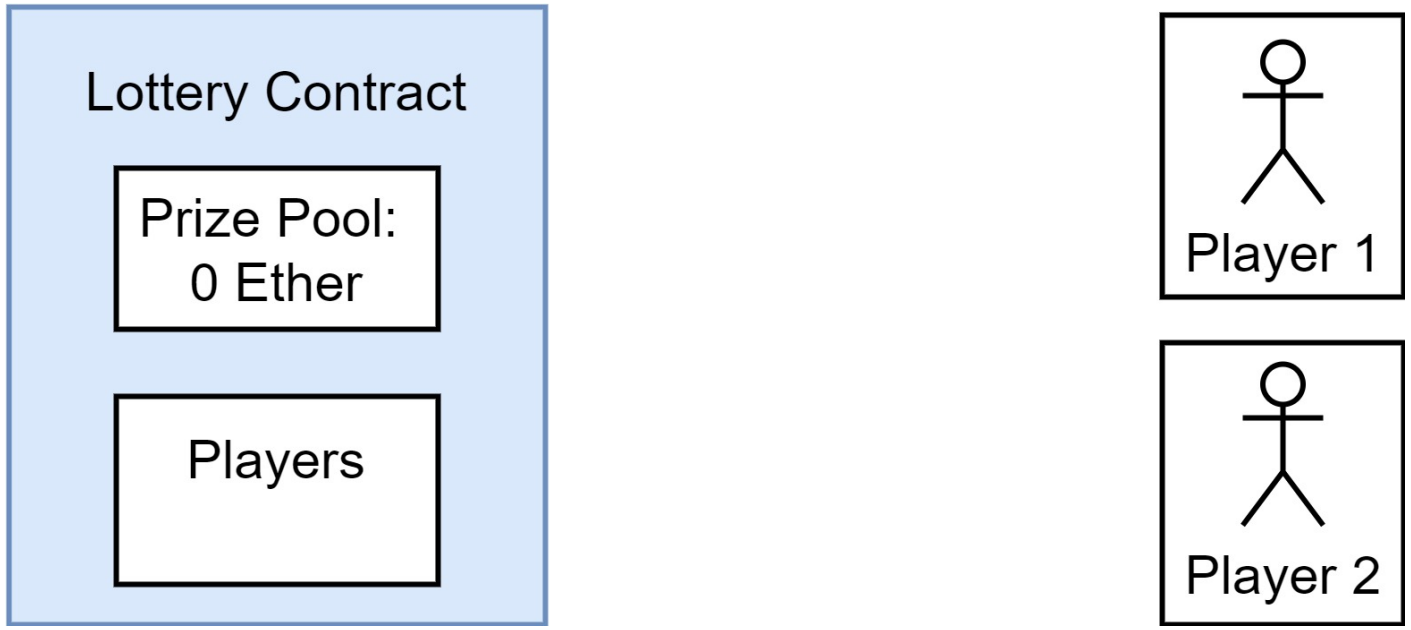
Blockchain for Industrial Engineers: Decentralized Application Development

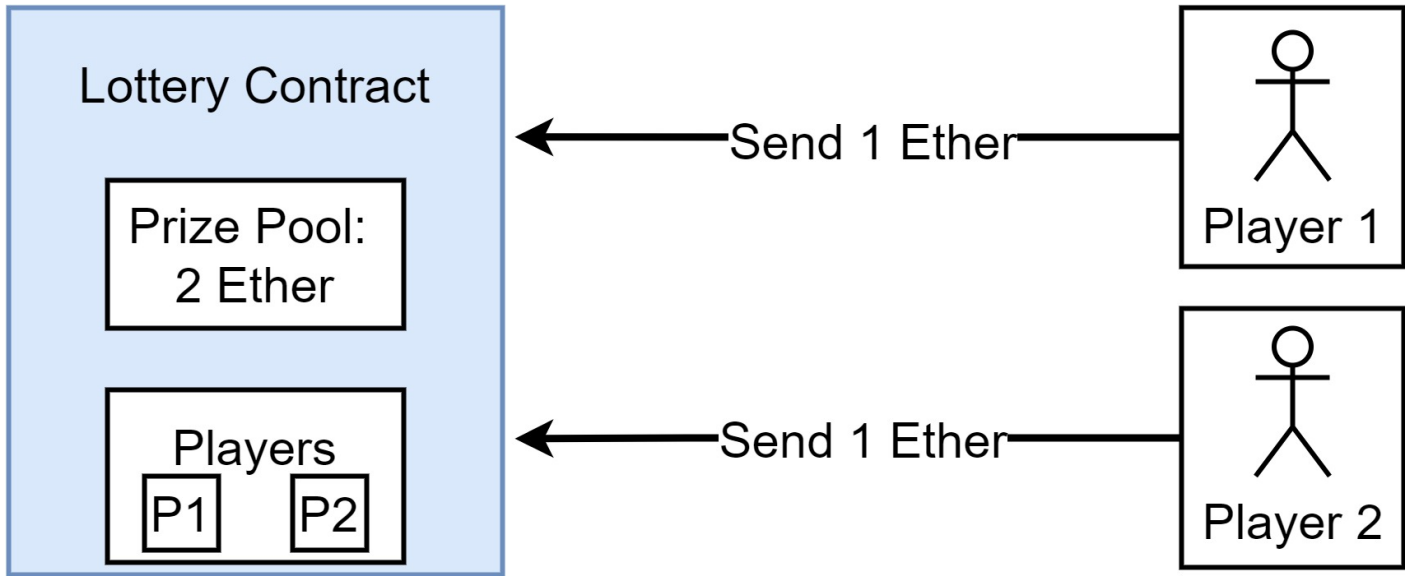
บล็อกเชนสำหรับวิศวกรอุตสาหการ: การพัฒนาแอปพลิเคชันแบบกระจายศูนย์

Lottery

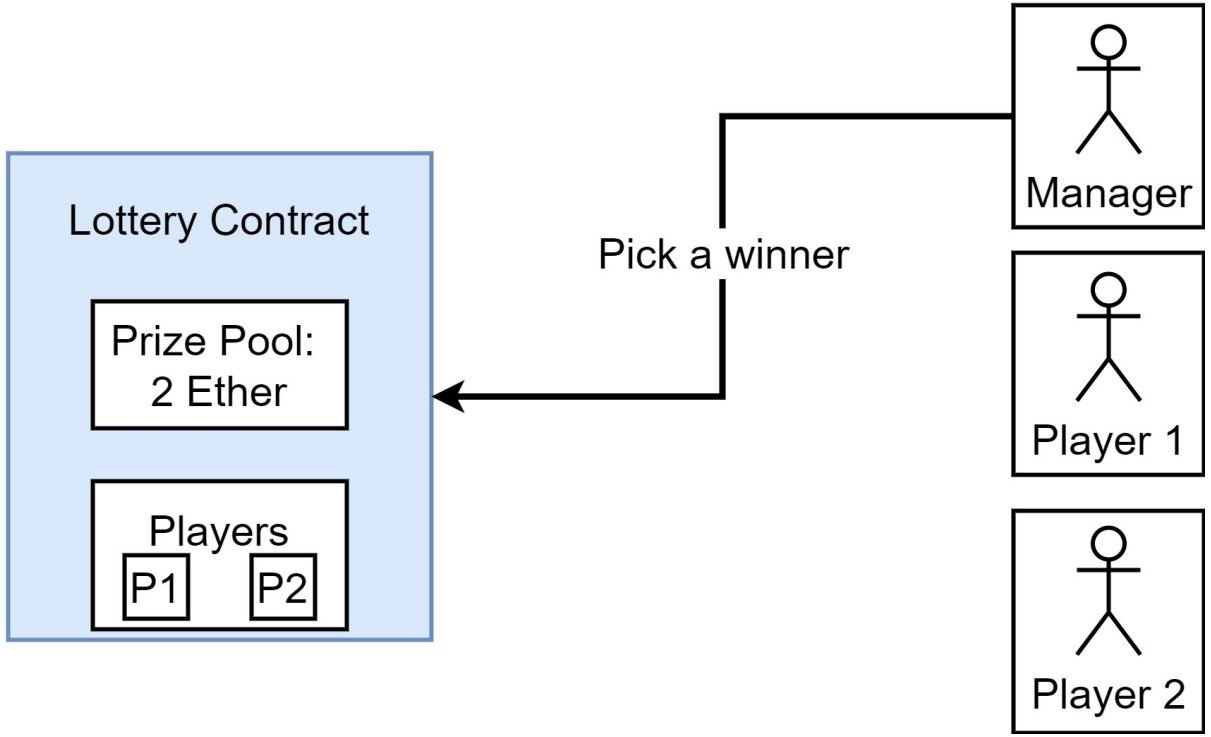
Setup


Lottery Contract
Prize Pool:
0 Ether
Players
Player 1
Player 2

Entering a lottery

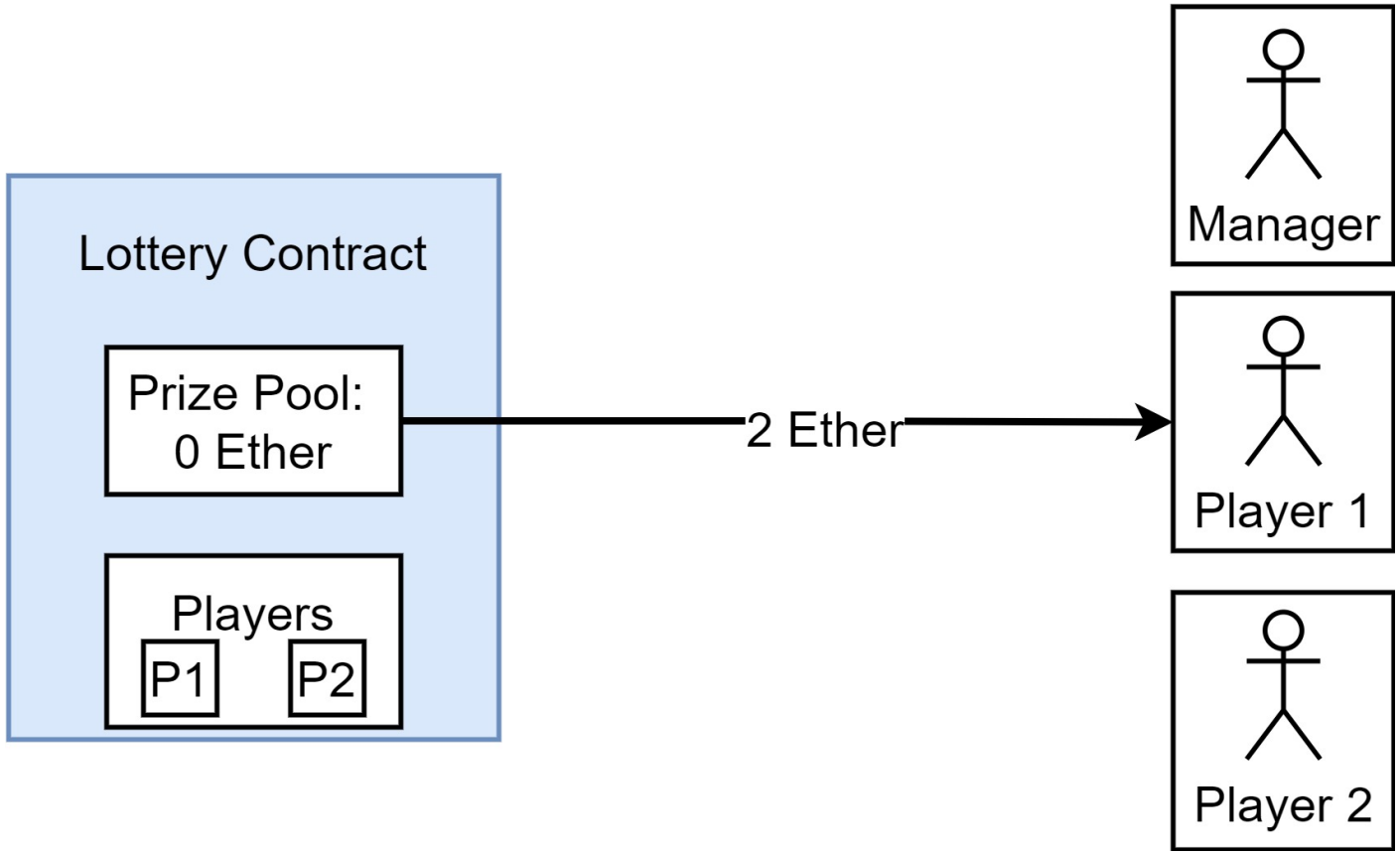


Picking a winner



Note: manager does not *choose* the winner, but only tell the contract to choose the winner.

Sending back the prize



Varaibles

Name	Purpose
`manager`	**Address** of person who created the contract
`players`	**Array** of addresses of people who have entered

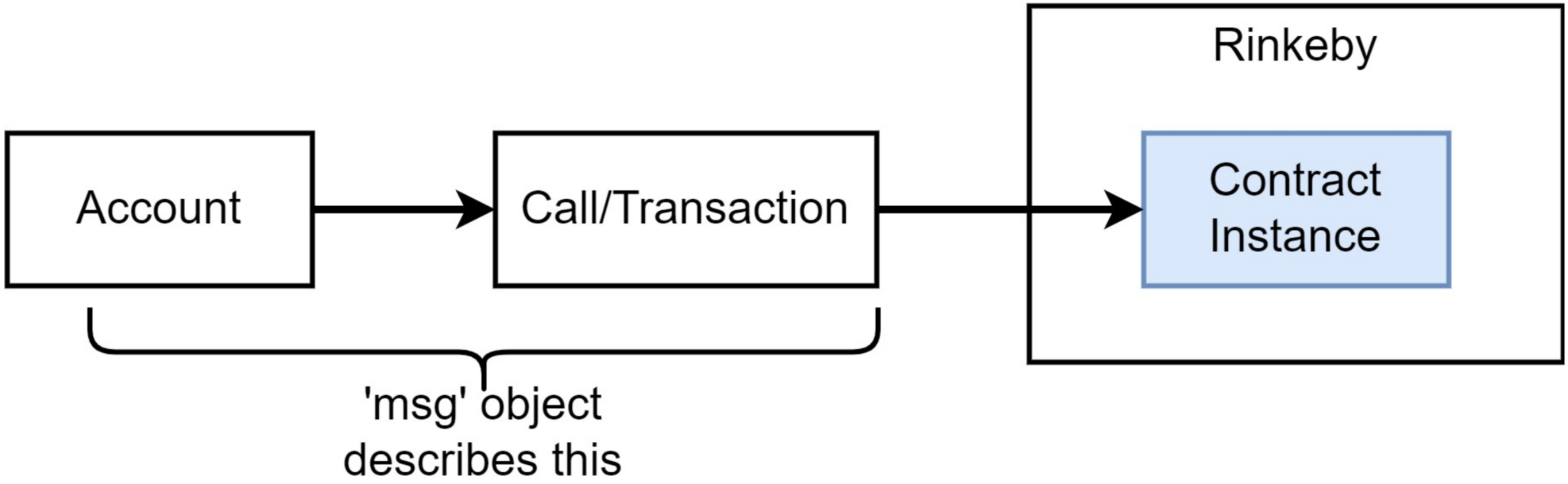
Functions

Name	Purpose
`enter`	Enters a player into the lottery
`pickWinner`	Randomly picks a winner and sends them the prize poolentered

Variables - `address`

Basic Types		
Name	**Notes**	**Examples**
string	Sequence of characters	"Hi there!" "Chocolate"
bool	Boolean value	true false
int	Integer, positive or negative. Has no decimal	0 -30000 59158
uint	'Unsigned' integer, positive number. Has no decimal	0 30000 999910
fixed/ufixed	'Fixed' point number. Number with a decimal after it	20.001 -42.4242 3.14
address	Has methods tied to it for sending money	0x18bae199c8dbae199c8d

`msg` object


Rinkeby
Account
Call/Transaction
Contract Instance
'msg' object describes this

The 'msg' Global Variable	
Property Name	Property Name
msg.data	'Data' field from the call or transaction that invoked the current function
msg.gas	Amount of gas the current function invocation has available
msg.sender	Address of the account that started the current function invocation
msg.value	Amount of ether (in wei) that was sent along with the function invocation

Storing `players`

- Array

Reference type varaibles

Reference Types		
Name	**Notes**	**Examples**
fixed array	Array that contains a *single type* of element. Has an unchanging length	`int[3] --> [1, 2, 3]` `bool[2] --> [true, false]`
dynamic array	Array that contains a *single type* of element. Can change in size over time	`int[] --> [1,2,3]` `bool[] --> [true, false]`
mapping	Collection of key value pairs. Think of Javascript objects, Ruby hashes, or Python dictionary. All keys must be of the same type, and all values must be of the same type	`mapping(string => string)` `mapping(int => bool)`
struct	Collection of key value pairs that can have different types.	`struct Car { string make; string model; uint value; }`