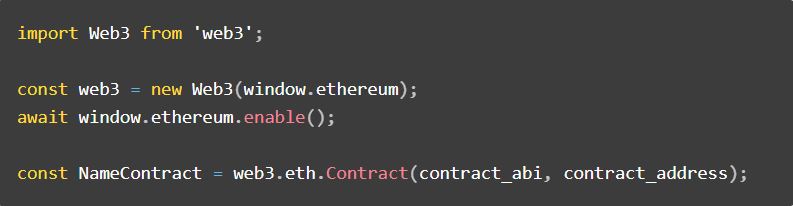
Web3.js is a library that allows you to do a number of things related to developing for the ethereum ecosystem, including interacting with a smart contract from a frontend application.

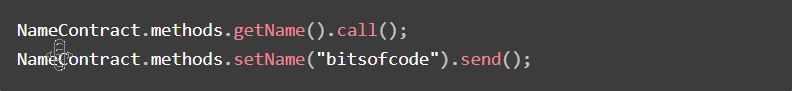
Let’s say we have a simple smart contract that allows you to do just two things - get a string variable **name**, and set that **name** variable to a new string.



In order to interact with this smart contract via a frontend application, we would first initialise web3.js with the smart contract ABI and address.



Once initialised, we can call any of the methods of our smart contract using either the **call()** or **send()** methods.

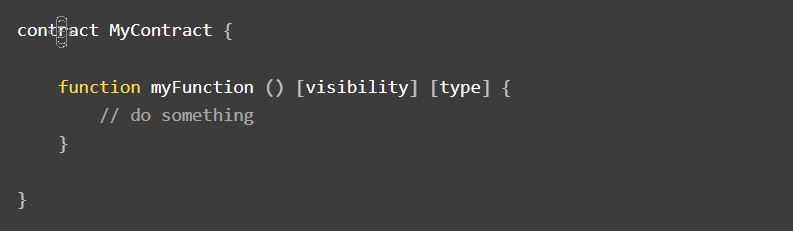


What’s the difference between call() and send()?

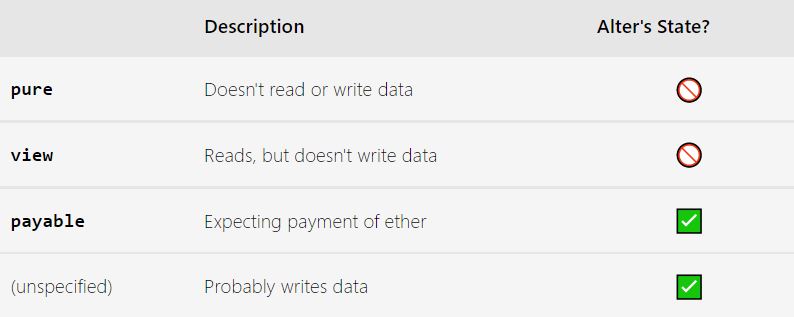
The difference between the **call()** and **send()** methods has to do with the type of function they are calling and their effect. Solidity functions can be divided into two categories:

* Functions that **alter** the state of the contract
* Functions that **do not alter** the state of the contract

To make it clear which category a function belongs to, we can specify the function type.



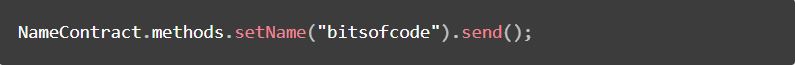
There are 4 function type keywords we can use, including leaving the function type unspecified.



Going back to my example contract, we can see that the function **getName()** has the type **view** which means it does not change the state of the contract. When calling this function via web3.js, we should use the **call()** method.



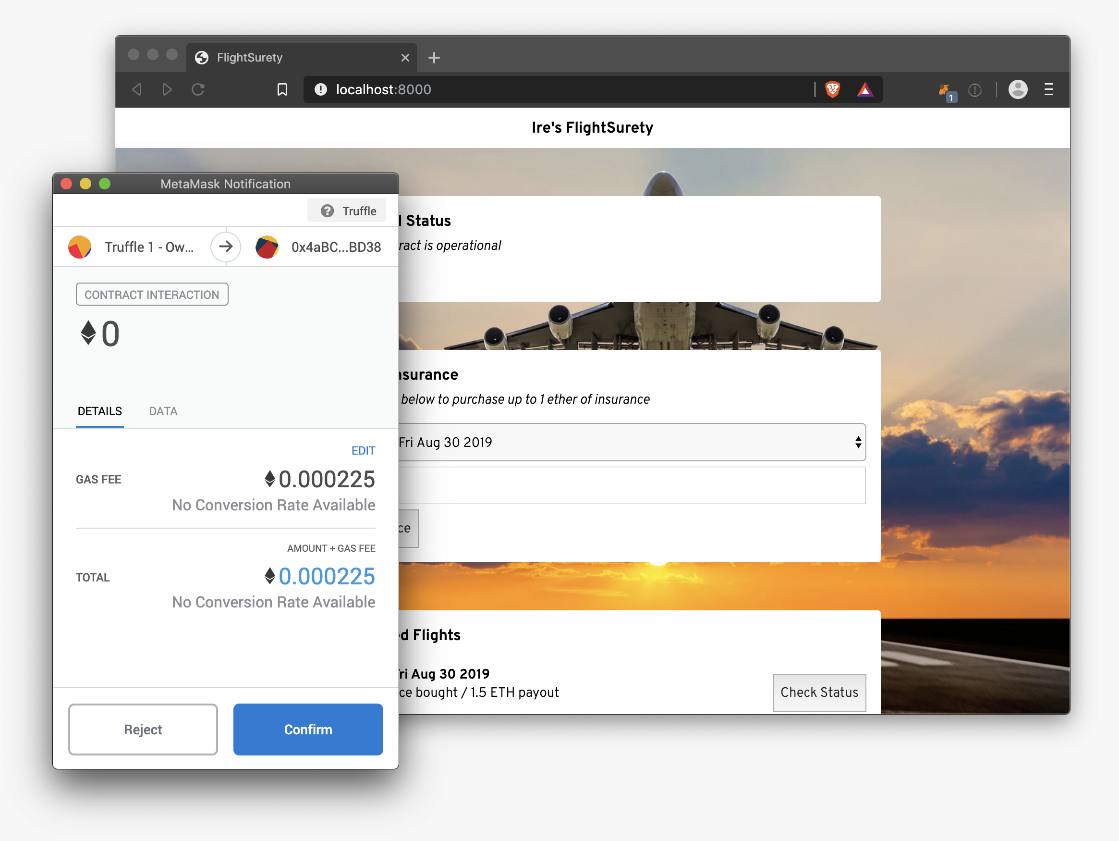
On the contrary, the **setName()** function doesn’t have a specified type and does change the state of the contract, in this case the value of the **name** variable. For these types of funtions, we should use the **send()** method.



What’s the effect of call() vs send()?

The reason we have two separate methods for calling functions that do or do not alter the state of a smart contract is because the former requires a transaction on the blockchain and the expense of ether in order to be effective.

The ether required to execute a smart contract function is called **“gas”** and it is a transaction that the user of the application will have to accept. If the user is using an ethereum wallet such as the Metamask plugin in their browser, they will have to accept the expense of the transaction via a popup window.



With functions called via the **call()** method on the other hand, they can happen silently in the background and do not require user input to complete.



What happens if you use the wrong method?

The reason this difference is such a big gotcha is because there is no explicit warning if you are using the wrong method to call your smart contract function.

For example, if I were to use the **call()** method to execute the **setName()** function, I would get no warnings.



Ostensibly, the method would have successfully completed with no errors. However, if we were to query the **name** variable that is supposed to have changed, we will see that it was not changed.

