

# Contract

When you instantiate an instance of `Contract` you need to specify the names of the functions you have on your smart contract. Then the new instance of `Contract` will have methods with the same names as your smart contract functions. For example if you deployed a contract with `my_smart_contract_function` function on it, then this will work:

```
const contract =  
new  
Contract ( account ,  
"example-contract.testnet" ,  
{ changeMethods :  
[ "my_smart_contract_function" ] ,  
// your smart-contract has a function my_smart_contract_function } ) ; //contract object has my_smart_contract_function function on it:  
contract . my_smart_contract_function ( ) ;
```

## Load Contract

- Standard
- Using Wallet

```
const  
{  
Contract  
}  
= nearAPI ;  
const contract =  
new  
Contract ( account ,  
// the account object that is connecting "example-contract.testnet" , { // name of contract you're connecting to viewMethods :  
[ "getMessages" ] ,  
// view methods do not change state but usually return a value changeMethods :  
[ "addMessage" ] ,  
// change methods modify state } ) ; ClassContract const  
{  
Contract  
}  
= nearAPI ;  
const contract =  
new  
Contract ( wallet . account ( ) ,  
// the account object that is connecting "example-contract.testnet" , { // name of contract you're connecting to viewMethods :  
[ "getMessages" ] ,  
// view methods do not change state but usually return a value changeMethods :
```

```
// change methods modify state } ) ;ClassContract
```

- Change Method
- Change Method w/ callbackUrl and meta
- View Method
- View Method w/ args

```
const contract =  
new  
Contract ( account ,  
  
"example-contract.testnet" ,  
  
{ changeMethods :  
  
[ "method_name" ] , } ) ; await contract . method_name ( { arg_name :  
  
"value" ,  
  
// argument name and value - pass empty object if no args required } , "3000000000000000" ,  
  
// attached GAS (optional) "1000000000000000000000000"  
  
// attached deposit in yoctoNEAR (optional) ) ; const contract =  
new  
Contract ( account ,  
  
"example-contract.testnet" ,  
  
{ changeMethods :  
  
[ "method_name" ] , } ) ; await contract . method_name ( { callbackUrl :  
  
"https://example.com/callback" ,  
  
// callbackUrl after the transaction approved (optional) meta :  
  
"some info" ,  
  
// meta information NEAR Wallet will send back to the application.meta will be attached to thecallbackUrl as a url param args :  
  
{ arg_name :  
  
"value" ,  
  
// argument name and value - pass empty object if no args required } , gas :  
  
3000000000000000 ,  
  
// attached GAS (optional) amount :  
  
1000000000000000000000000 ,  
  
// attached deposit in yoctoNEAR (optional) } ) ; const contract =  
new  
Contract ( account ,  
  
"example-contract.testnet" ,  
  
{ viewMethods :  
  
[ "view_method_name" ] , } ) ; const response =  
  
await contract . view  method  name ( ) ; const contract =
```

new

Contract ( account ,

"example-contract.testnet" ,

{ viewMethods :

[ "view\_method\_name" ] , } ) ; const response =

await contract . view\_method\_name ( {

arg\_name :

"arg\_value"

} ) ; [ClassContract](#) [Edit this page](#) Last updated on Jan 31, 2024 by gagdiez Was this page helpful? Yes No

[Previous Account](#) [Next Utilities](#)