XRC20 & XRC721 Token Standard

XRC20 Token Standard

What is an XRC20 token?

- An XRC20 token is a blockchain-based asset with similar functionality to ether, it can hold value and be sent and received.
- XRC20 tokens are stored and sent using Xinfin addresses and transactions, and use gas to cover transaction fees.

XRC20 Token: Read methods	XRC20 Token: Write methods
name()	transfer(recipient, amount)
symbol()	approve(spender, amount)
decimal()	transferFrom(sender, recipient, amount)
totalSupply()	increaseAllowance(spender, addedValue)
balanceOf(account)	decreaseAllowance(spender, subtractedValue)
allowance(owner, spender)	

`name()  string`
Returns the name of the token.

`symbol()  string`
Returns the symbol of the token, usually a shorter version of the name.

`decimals()  uint8`
Returns the number of decimals used to get its user representation. For example, if decimals equals 2, a balance of 505 tokens should be displayed to a user as 5,05 (505 / 10 ** 2).

Tokens usually opt for a value of 18, imitating the relationship between Ether and Wei.

`totalSupply()  uint256`
Returns the number of tokens in existence.

`balanceOf(address account)  uint256`
Returns the number of tokens owned by the account.

`allowance(address owner, address spender)  uint256`
Returns the remaining number of tokens that the spender will be allowed to spend on behalf of the owner through transferFrom. This is zero by default.

`approve(address spender, uint256 amount)`
Sets amount as the allowance of spender over the caller’s tokens.

`transfer(address recipient, uint256 amount)`
Moves amount tokens from the caller’s account to the recipient. It helps in moving number of tokens from one account to another.

`transferFrom(address sender, address recipient, uint256 amount)`
Moves amount tokens from sender to recipient using the allowance mechanism. the amount is then deducted from the caller’s allowance.

`increaseAllowance(address spender, uint256 addedValue)`
Automically increases the allowance granted to spender by the caller.

`decreaseAllowance(address spender, uint256 addedValue)`
Automically decreases the allowance granted to spender by the caller.

XRC20 Methods

Usage

```
Usage
```

```
When the user wants to use the SDKPHP library he/ she need to follow
the below steps
Step1 :- create a folder User convient (eg : sdk)
Step2 :- inside the folder open the command prompt and type composer
require xdc3/php
Step3 :- create a file according to  User convient (eg : index.php)
Step4 :- After creating user need to add the path , import the classes
and create the object
Step5 :- (oprional) If user wants to change the URL  Goto vendor ->
XDC3 folder -> PHP folder -> .env file

Important :

 Important :

This library provides simple way to interact with XDC XRC20 & XRC721
tokens.


Before Installing This Library

1.Install php version 7.4.25 in the System.



2.go to PHP Folder in the system, open php.ini file Then enable gmp and
curl extensions.

ex: before it has ;extension=gmp just remove ";" to enable this
Extension.

    before it has ;extension=curl just remove ";" to enable this
Extension.



3.after that enable this one(;curl.cainfo =) and add ssl
certificatepath here.

  for download this certificate

  go to this url https://curl.se/docs/caextract.html then click on
cacert.pem on the top.

  then copy the path of the certificate and paste in php .ini file.

  ex: curl.cainfo = "C:\Users\HP\Downloads\cacert.pem"
```

```
4.Install openssl in the system by chocolately.

  Run below command in admin powershell to install openssl.

  ex: choco install openssl


 5. -In XDC library inside the vendor folder -> xdc3base -> php  we can
find the all called functions of the XRC20 methods . (including
contract file , address calling from apothem file etc)

    In PHPS library inside the vendor folder -> xdc3 -> php  we can
find the all calling functions from the XDC libarary. (ex : in modules
xrc20 folder contains every calling methods )

//Folder Info //
-In XDC library inside the vendor folder -> xdc3base -> php  we can
find the all called functions of the XRC20 methods . (including
contract file , address calling from apothem file etc)

In PHPS library inside the vendor folder -> xdc3 -> php  we can find
the all calling functions from the XDC libarary. (ex : in modules xrc20
folder contains every calling methods )

If user wants to change the URL  Goto vendor -> XDC3 folder -> PHP
folder -> .env file   .

ex :-//path configurationinclude "vendor/xdc3/php/library.php";//Import
classes of XDC20 use XRC721\SDK\XRC20SDK;

//Creating a Objects to call the functions of XRC20 and Create a wallet
for the User$obj1 = new XRC20SDK();$obj3 = new createXDCWallet();
```

ex :-
//path configuration
include "vendor/phps/sdkxdc/library.php";
//Import classes of XDC20
`use XRC721\SDK\XRC20SDK;`

//Creating a Objects to call the functions of XRC20 and Create a wallet for the User
$obj1 = new XRC20SDK();
$obj3 = new createXDCWallet();

Test.php

```
<?php
//path configuration
include "vendor/phps/sdkxdc/library.php";

//Import classes of XDC20
use XRC20\SDK\XRC20SDK;

//Creating a Objects to call the functions of XRC20 and Create a wallet for the User
$obj1 = new XRC20SDK();
```

Name func :-
`$obj1->getName($contractAddress);`

Symbol func :-

```
  $obj1->getSymbol($contractAddress);
```

Decimal func :-

```
$obj1->getDecimal($contractAddress);
```

TotalSupply func :-

```
$obj1->getTotalSupply($contractAddress);
```

BalanceOf func :-

```
$obj1->getBalanceOf($contractAddress,$ownerAddress);
```

Allowance func:-

```
$obj1->getAllowance($contractAddress,$ownerAddress,$spenderAddress);
```

GetApprove func:-

```
$obj1->getApprove($contractAddress,$ownerAddress,$ownerPrivateKey,$spenderAddress,$tokenAmount);;
```

IncreaseAllowance func :-

```
    $obj1-

>increaseAllowance($contractAddress,$ownerAddress,$ownerPrivateKey,$spenderAddress,$tokenAmount);
```

DecreaseAllowance func :-

```
    $obj1->decreaseAllowance($contractAddress,$ownerAddress,$ownerPrivateKey,$spenderAddress,$tokenAmount);
```

TransferFrom func :-

```
 $obj1->transferFrom
($contractAddress,$ownerAddress,$spenderAddress,$spenderPrivateKey,$recieverAddress,$tokenAmount);
```

TransferToken func :-

```
$obj1->transferToken($contractAddress,$senderAddress,$senderPrivateKey,$recieverAddress,$tokenAmount);
```

TransferXdc func :-

```
$obj1->transferXdc($contractAddress,$senderAddress,$senderPrivateKey,$recieverAddress,$xdcAmount);
```

CreateWallet func:-

```
$obj3 = new createXDCWallet();

$obj3->createWallet();
?>
```

XRC721 Token Standard

What is an XRC721 token?

· The XRC-721 introduces a standard for NFT, in other words, this type of Token is unique and can have different value than another Token from the same Smart Contract, maybe due to its age, rarity or even something else like its visual. Wait, visual?

Yes! All NFTs have a uint256 variable called tokenId, so for any XRC-721 Contract, the pair contract address, uint256 tokenId must be globally unique.

Every XRC-721 compliant contract must implement the `XRC721` and `XRC165` interfaces.

XRC721 Token: Read methods	**XRC Token: Write methods**
name()	transferFrom(recipient, amount)
symbol()	safeTransferFrom(spender, amount)
totalSupply()	setApprovalForAll(operatorAddress, booleanValue)
balanceOf(owner address)	approve(receiverAddress , tokenId)
ownerOf(tokenId)	
tokenURI(tokenId)	
tokenByIndex(index)	
tokenOfOwnerByIndex(ownerAddress,index)	

supportInterface(interfaceId)	
getApproved(tokenId)	
isApprovedForAll(ownerAddress, operatorAddress)	

`name()  string`
Returns the name of the token.

`symbol()  string`
Returns the symbol of the token, usually a shorter version of the name.

`totalSupply()  uint256`
Gets the total amount of tokens stored by the contract.

`balanceOf(address account)  uint256`
Returns the number of NFTs in `owner`'s account.

`ownerOf(tokenId)  address`
Returns the owner of the NFT specified by `tokenId`.

`tokenURI(uint256 tokenId)  string`
Returns the URI for a given token ID. May return an empty string.

`tokenByIndex(uint256 index)  uint256`
Gets the token ID at a given index of all the tokens in this contract Reverts if the index is greater or equal to the total number of tokens.

`tokenOfOwnerByIndex(address owner, uint256 index)  uint256`
Gets the token ID at a given index of the tokens list of the requested owner.

`supportInterface(tokenId)  bool`
Returns true if this contract implements the interface defined by `interfaceId`.

`getApproved(tokenId)  address`
Gets the approved address for a token ID, or zero if no address set Reverts if the token ID does not exist.

`isApprovedForAll (address owner, address operator)  bool`
Tells whether an operator is approved by a given owner.

`transferFrom(address from, address to, uint256 tokenId)`
Transfers the ownership of a given token ID to another address.

`safeTransferFrom(address from, address to, uint256 tokenId)`
Safely transfers the ownership of a given token ID to another address

`setApprovalForAll(address operator, bool approved)`
Sets or unsets the approval of a given operator An operator is allowed to transfer all tokens of the sender on their behalf.

`approve(address to, uint256 tokenId)`
Approves another address to transfer the given token ID The zero address indicates there is no approved address. There can only be one approved address per token at a given time. Can only be called by the token owner or an approved operator.

XRC721 Methods

Usage

```
Usage

When the user wants to use the SDKPHP library he/ she need to follow
the below steps
Step1 :- create a folder User convient (eg : sdk)
Step2 :- inside the folder open the command prompt and type composer
require xdc3/php
Step3 :- create a file according to  User convient (eg : index.php)
Step4 :- After creating user need to add the path , import the classes
```

```
and create the object
Step5 :- (oprional) If user wants to change the URL  Goto vendor ->
XDC3 folder -> PHP folder -> .env file

Important :

This library provides simple way to interact with XDC XRC20 & XRC721
tokens.


Before Installing This Library

1.Install php version 7.4.25 in the System.



2.go to PHP Folder in the system, open php.ini file Then enable gmp and
curl extensions.

ex: before it has ;extension=gmp just remove ";" to enable this
Extension.

    before it has ;extension=curl just remove ";" to enable this
Extension.



3.after that enable this one(;curl.cainfo =) and add ssl
certificatepath here.

  for download this certificate

  go to this url https://curl.se/docs/caextract.html then click on
cacert.pem on the top.

  then copy the path of the certificate and paste in php .ini file.

  ex: curl.cainfo = "C:\Users\HP\Downloads\cacert.pem"



4.Install openssl in the system by chocolately.

  Run below command in admin powershell to install openssl.

  ex: choco install openssl

  //Folder Info //
  -In XDC library inside the vendor folder -> xdc3base -> php  we can
find the all called functions of the XRC20 methods . (including
contract file , address calling from apothem file etc)

    In PHPS library inside the vendor folder -> xdc3 -> php  we can
find the all calling functions from the XDC libarary. (ex : in modules
xrc20 folder contains every calling methods )
```

ex :-
//path configuration
include "vendor/phps/sdkxdc/library.php";
//Import classes of XDC721

```
use XRC721\SDK\XRC721SDK;
```

//Creating a Objects to call the functions of XRC20 and Create a wallet for the User
$obj2 = new XRC721SDK();

Name func :-

```
$obj2->getName($contractAddress);
```

Symbol func :-

```
  $obj2->getSymbol($contractAddress);
```

TotalSupply func :-

```
$obj2->getTotalSupply($contractAddress);
```

BalanceOf func :-

```
$obj2->getBalanceOf($contractAddress,$ownerAddress);
```

SupportsInterface func :-

```
$obj2->getSupportInterface($contractAddress,$interfaceId);
```

OwnerOf func :-

```
$obj2->getOwnerOf($contractAddress,$tokenId);
```

TokenURI func:-

```
$obj2->getTokenURI($contractAddress,$tokenId);
```

TokenByIndex func :-

```
$obj2->getTokenByIndex($contractAddress,$index);
```

TokenOfOwnerByIndex func :-

```
$obj2->getTokenOfOwnerByIndex($contractAddress,$ownerAddress,$index);
```

GetApprove func:-

```
$obj2->getApproved($contractAddress,$tokenId);
```

Approve func:-

```
$obj2->approve($contractAddress,$recieverAddress,$tokenId,$privateKey);
```

SetApprovedForAll func :-

```
$obj2->setApprovalForAll($contractAddress,$operatorAddress,$approvedStatus,$ownerPrivateKey,$tokenId);
```

IsApprovedForAll func :-

```
$obj2->setApprovalForAll($contractAddress,$operatorAddress,$approvedStatus,$ownerPrivateKey,$tokenId);
```

SafeTransferFrom func:-

```
 $obj2->safeTransferFrom($contractAddress,$ownerAddress,$recieverAddress,$tokenId,$approvedPrivateKey);
```

TransferFrom func:-

```
 $obj2->transferFrom($contractAddress,$ownerAddress,$recieverAddress,$tokenId,$approvedPrivateKey);
```

TransferOwnership

```
 $obj2->transfer($contractAddress,$ownerAddress,$recieverAddress,$tokenId,$approvedPrivateKey);
```