Collateralized Debt Position

Note that this page applies only to Single-Collateral Sai.

After opening a new CDP or getting an existing one , you can use the returnedcdp object to call functions on it.

Properties

id

This is the ID of the CDP object. You can pass this ID to Maker.getCdp .

Methods

getDebtValue

- Params:
- currency unit
- (optional)
- Returns:
- promise (resolves to the amount of outstanding debt)
-

cdp.getDebtValue() returns the amount of debt that has been borrowed against the collateral in the CDP. By default it returns the amount of Dai as a currency unit , but can return the equivalent in USD if the first argument isMaker.USD .

```
Copy constdaiDebt=awaitcdp.getDebtValue(); constusdDebt=awaitcdp.getDebtValue(Maker.USD);
```

getGovernanceFee

- Params:
- currency unit
- (optional)
- Returns:
- promise (resolves to the value of the accrued governance fee in USD)
-

cdp.getGovernanceFee() returns the value of the accrued governance fee. By default it returns the amount of MKR as a currency unit , but can return the equivalent in USD if the first argument isMaker.USD .

Note: this is often referred to as theStability Fee , even though technically theStability Fee is the fee that is paid in Dai, and theGovernance Fee is the fee that is paid in MKR. But since fees are only paid in MKR in Single-Collateral Dai, and only paid in Dai in Multi-Collateral Dai, the fee in Single-Collateral Dai is often referred to as theStability Fee to be consistent with the term that will be used in Multi-Collateral Dai and to avoid unduly confusing regular users.

```
Copy constmkrFee=awaitcdp.getGovernanceFee(); constusdFee=awaitcdp.getGovernanceFee(Maker.USD);
```

getCollateralizationRatio

- Params:
- none
- Returns:
- promise (resolves to the collateralization ratio)
-

cdp.getCollateralizationRatio() returns the USD value of the collateral in the CDP divided by the USD value of the Dai debt for the CDP, e.g.2.5 .

```
Copy constratio=awaitcdp.getCollateralizationRatio();
```

```

getLiquidationPrice

- Params:
- none
- Returns:
- promise (resolves to the liquidation price)
- 

cdp.getLiquidationPrice() returns the price of Ether in USD that causes the CDP to become unsafe (able to be liquidated), all other factors constant. It returns aUSD_ETH price unit .

```

Copy constratio=awaitcdp.getLiquidationPrice();

```

getCollateralValue

- Params:
- currency unit
- (optional)
- Returns:
- promise (resolves to collateral amount)
- 

cdp.getCollateralValue() returns the value of the collateral in the CDP. By default it returns the amount of ETH as acurrency unit , but can return the equivalent in PETH or USD depending on the first argument.

```

Copy constethCollateral=awaitcdp.getCollateralValue(); constpethCollateral=awaitcdp.getCollateralValue(Maker.PETH); constusdCollateral=awaitcdp.getCollateralValue(Maker.USD);

```

isSafe

- Params:
- none
- Returns:
- promise (resolves to boolean)
- 

cdp.isSafe() returns true if the cdp is safe, that is, if the USD value of its collateral is greater than or equal to USD value of the its debt multiplied by the liquidation ratio.

```

Copy constratio=awaitcdp.isSafe();

```

enoughMkrToWipe

- Params:
- amount of Sai to wipe
- Returns:
- promise (resolves to boolean)
- 

cdp.enoughMkrToWipe(dai) returns true if the current account owns enough MKR to wipe the specified amount of Sai from the CDP.

```

Copy constenoughMkrToWipe=awaitcdp.enoughMkrToWipe(10000000000000000000,SAI.wei);

```
```

lockEth

- Params:
- amount to lock in the CDP, in units defined by the price service.
- Returns:
- promise (resolves totransactionObject
- once mined)
-

cdp.lockEth(eth) abstracts the token conversions needed to lock collateral in a CDP. It first converts the ETH to WETH, then converts the WETH to PETH, then locks the PETH in the CDP.

Note: this process is not atomic, so it's possible for some of the transactions to succeed but not all three. SeeUsing DsProxy for executing multiple transactions atomically.

```

Copy returnawaitcdp.lockEth(100000000000000000000,ETH.wei); // or equivalently returnawaitcdp.lockEth(100,ETH);

```

drawSai

- Params:
- amount to draw (in Sai, as string)
- Returns:
- promise (resolves totransactionObject
- once mined)
-

cdp.drawSai(sai) withdraws the specified amount of Sai as a loan against the collateral in the CDP. As such, it will fail if the CDP doesn't have enough PETH locked in it to remain at least 150% collateralized.

```

Copy returnawaitcdp.drawSai(100000000000000000000,SAI.wei); // or equivalently returnawaitcdp.drawSai(100,SAI);

```

wipeSai

- Params:
- amount to repay (in Sai, as string)
- Returns:
- promise (resolves totransactionObject
- once mined)
-

cdp.wipeSai(sai) sends Sai back to the CDP in order to repay some (or all) of its outstanding debt.

Note: CDPs accumulate MKR governance debt over their lifetime. This must be paid when wiping sai debt, and thus MKR must be acquired before calling this method.

```

Copy returnawaitcdp.wipeSai(100000000000000000000,SAI.wei); // or equivalently returnawaitcdp.wipeSai(100,SAI);

```

freePeth

- Params:
- amount of Peth collateral to free from the CDP, in units defined by the price service.
- Returns:
- promise (resolves totransactionObject
- once mined)
-

cdp.freePeth(peth) withdraws the specified amount of PETH and returns it to the owner's address. As such, the contract will only allow you to free PETH that's locked in excess of 150% of the CDP's outstanding debt.

```
```

Copy returnawaitcdp.freePeth(100,PETH); // or equivalently returnawaitcdp.freePeth(100000000000000000000,PETH.wei);

```

give

- Params:
- Ethereum address (string)
- Returns:
- promise (resolves totransactionObject
- once mined)
-

cdp.give(address) transfers ownership of the CDP from the current owner to the address you provide as an argument.

```

Copy returnawaitcdp.give('0x046ce6b8ecb159645d3a605051ee37ba93b6efcc');

```

shut

- Params:
- none
- Returns:
- promise (resolves totransactionObject
- once mined)
-

cdp.shut() wipes all remaining sai, frees all remaining collateral, and deletes the CDP. This will fail if the caller does not have enough DAI to wipe all the sai debt and enough MKR to pay for all the accrued stability fee

```

Copy returnawaitcdp.shut();

```

bite

- Params:
- none
- Returns:
- promise (resolves totransactionObject
- once mined)
-

cdp.bite() will initiate the liquidation process of an undercollateralized CDP

```

Copy returnawaitcdp.bite();

```

Previous Single-Collateral Sai Next CDP Service Last updated3 years ago On this page *Properties * id * Methods * getDebtValue * getGovernanceFee * getCollateralizationRatio * getLiquidationPrice * getCollateralValue * isSafe * enoughMkrToWipe * lockEth * drawSai * wipeSai * freePeth * give * shut * bite
```