スマートコントラクトと

メタトランザクションについて

2023年7月15日

目次

JPYC

JPYCv2

- JPYCv2
  - centreに準拠
  - Blocklist
  - Upgradeability
  - EIP2612 / 3009（メタトランザクション）
  - Pause
  - Rescuable
  - Minter

2. ERC20

ERC20とは

- Ethereum（EVM）におけるトークンの共通規格

```
function name() public view returns (string)
function symbol() public view returns (string)
function decimals() public view returns (uint8)
function totalSupply() public view returns (uint256)
function balanceOf(address _owner) public view returns (uint256 balance)
function transfer(address _to, uint256 _value) public returns (bool success)
function transferFrom(address _from, address _to, uint256 _value) public returns (bool success)
function approve(address _spender, uint256 _value) public returns (bool success)
function allowance(address _owner, address _spender) public view returns (uint256 remaining)
event Transfer(address indexed _from, address indexed _to, uint256 _value)
event Approval(address indexed _owner, address indexed _spender, uint256 _value)
```

ERC20のbalanceとtransfer

balance (残高)

```
mapping(address => uint256) private _balances;
*key-value (ex. 社員番号)

function balanceOf(
        address account
) public view virtual override returns (uint256) {
        return _balances[account];
}
```

transfer (送信)

```
function _transfer(
        address from,
        address to,
        uint256 amount
) internal virtual {
        uint256 fromBalance = _balances[from];
        unchecked {
                _balances[from] = fromBalance -
                amount;
                _balances[to] += amount;
        }
}
```

ERC20の注意点

- ERC20は関数名と引数を定義しただけで、関数のロジックは規格がない
- いいケース
  - プログラマブルマネーを構築する
- 悪いケース
  - トークンに見せかけて悪意のあるコードを仕込む

ERC20のまとめ

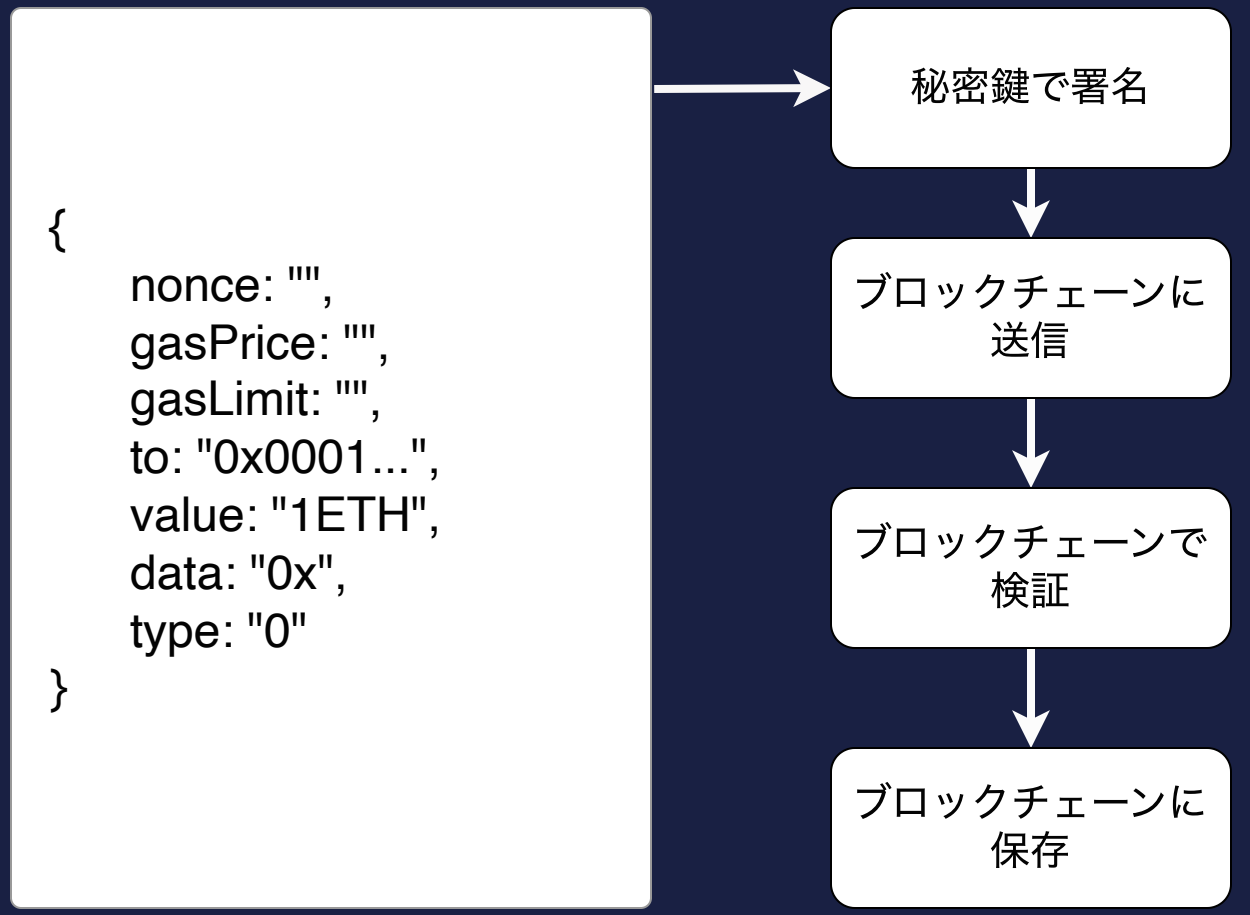
- トークンの共通の規格
- mappingでずべての残高を管理している
- インターフェースを定義しただけで、中のロジックは定義されていない

3. メタトランザクション

メタトランザクションとは

- トランザクションの署名と実行の分離

トランザクションの仕組み



トークンを使うにあたっての問題点

- ガス代がないとトークンを送信できない
- ガス代のトークンを調達するのが手間
- ガス代の価格変動のリスクに晒される
- 暗号資産の管理・保有が大変

メタトランザクションとは

- トランザクションの署名と実行の分離
- トランザクションを誰かに肩代わりしてもらい、ブロックチェーン上の操作を行うこと

EIP 2612 / EIP 3009

- EIP 2612
  - 署名でapproveが実行できるもの
- EIP 3009
  - 署名でtransferが実行できるもの

EIP 3009 と 通常とERC20のTransfer

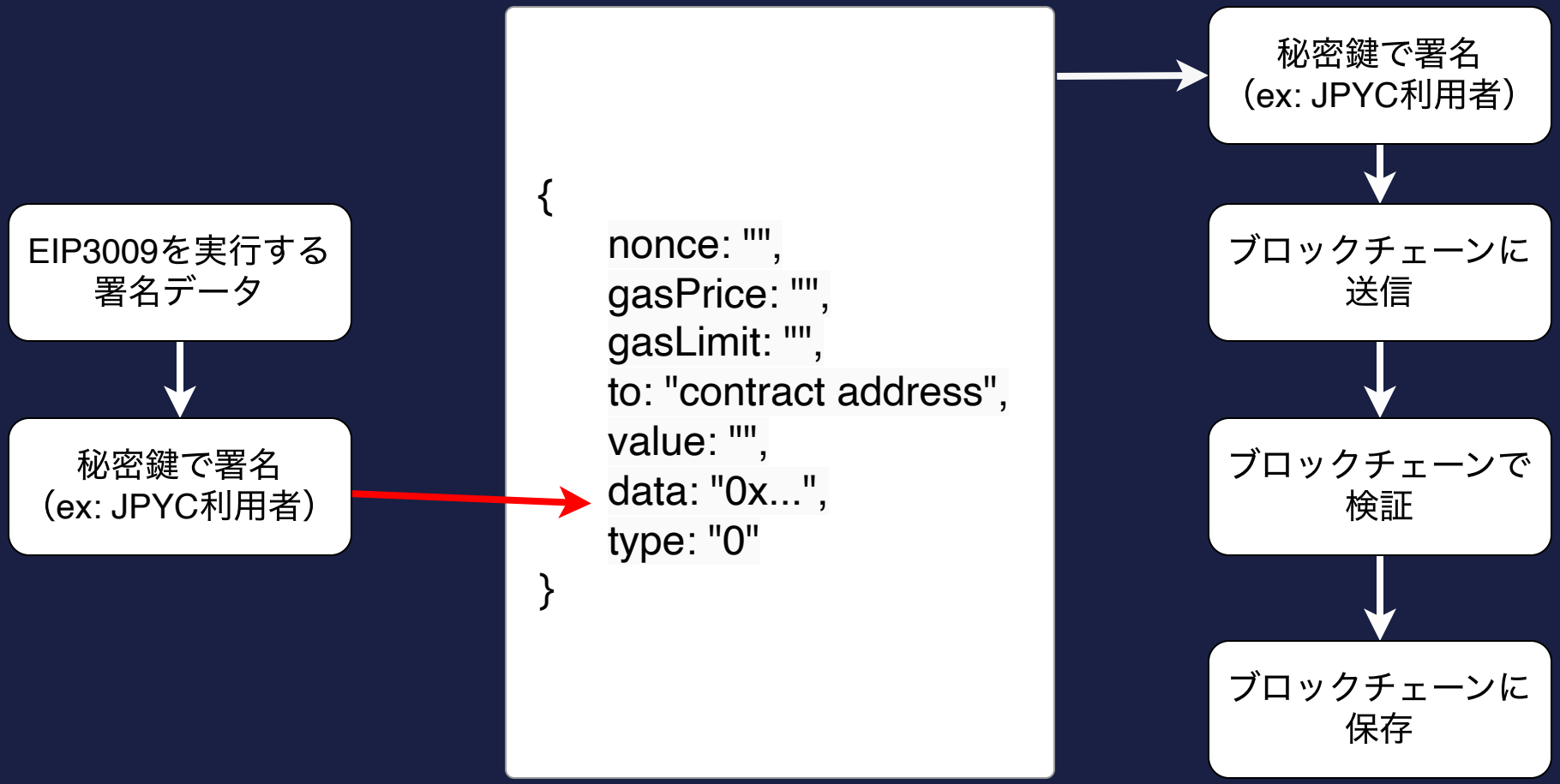
```
function _transferWithAuthorization(
        address from, address to, uint256 value,
        uint256 validAfter, uint256 validBefore,
        bytes32 nonce, uint8 v, bytes32 r, bytes32 s
) internal {
        _requireValidAuthorization(from, nonce, validAfter, validBefore);

        bytes memory data = abi.encode(
        TRANSFER_WITH_AUTHORIZATION_TYPEHASH,
                from, to, value,
                validAfter, validBefore, nonce
        );
        require(
                EIP712.recover(_domainSeparatorV4(), v, r, s, data) == from,
                "EIP3009: invalid signature"
        );

        _markAuthorizationAsUsed(from, nonce);
        _transfer( from , to, value);
}
```

```
function transfer(address to, uint256 value)
        external
        override
        whenNotPaused
        notBlocklisted(msg.sender)
        notBlocklisted(to)
        returns (bool)
{
        _transfer( msg.sender , to, value);
        return true;
}
```

EIP 3009


EIP3009を実行する
署名データ
秘密鍵で署名
（ex: JPYC利用者）
{
nonce: "",
gasPrice: "",
gasLimit: "",
to: "contract address",
value: "",
data: "0x...",
type: "0"
}
秘密鍵で署名
（ex: JPYC利用者）
ブロックチェーンに
送信
ブロックチェーンで
検証
ブロックチェーンに
保存

メタトランザクションまとめ

- 署名するデータと署名データから署名したアドレスが復元できる
- ERC20の規格に準拠しスマートコントラクトの範囲内で署名と実行の分離を行っている

4. ハッカソンに向けて

開発者としてブロックチェーンをどう捉えるか

- Web2 : 情報革命 → 完成系としてのAI
- Web3 : 価値革命 →　??
  - **ブロックチェーンではそれ以外の分野（何かしらの価値がつくもの）も大きく変える**

開発者としてブロックチェーンをどう捉えるか

- ブロックチェーン = Ethereumはstate machine ≒ DB
- スマートコントラクト = stateの書き込みと読み込みの定義 ≒ API
  - お金を払えば誰でも書き込みができる、読み込みはタダでできる
  - その書き込まれたデータは恣意的に変更できない（セキュリティー）

価値がつかなかったものに価値をつける

流動性の低かった（市場に出てなかった）資産の流動性を向上させる