Praxissession 4 - Grundlagen Dapp Entwicklung

Web-Applikationen mit Node.js und Express

1. Öffnen Sie ein neues Terminalfenster.
2. Neuen Ordner anlegen:

Terminalfenster / Konsole

```
$ mkdir mydapps
```

3. In den neuen Ordner wechseln:

```
$ cd mydapps
```

4. Ein neues JavaScript Package Repository anlegen.

```
$ npm init -y
```

5. JavaScript-Tool **express-generator** installieren:

```
$ npm install express-generator
```

6. Grundgerüst der Webseite anlegen:

```
$ ./node_modules/.bin/express --view=pug firstdapp
```

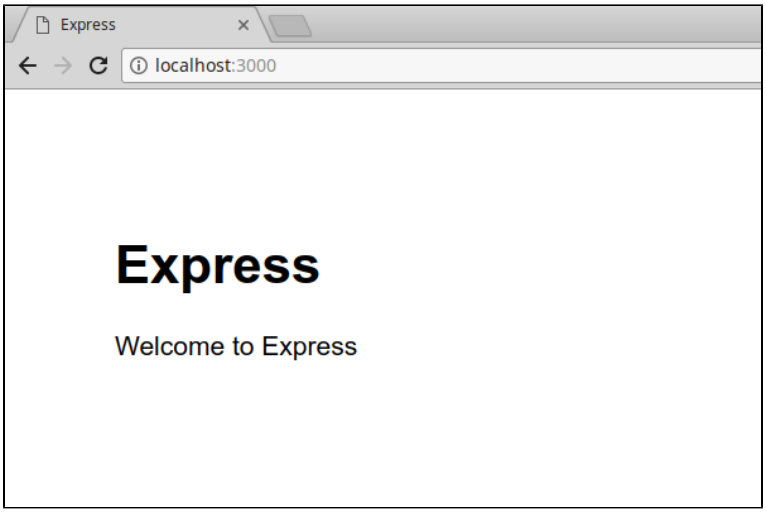
7. In den Ordner der neuen Webseite wechseln und die benötigten Dependecies installieren:

```
$ cd firstdapp
$ npm install
$ npm install web3
```

8. Webserver starten:

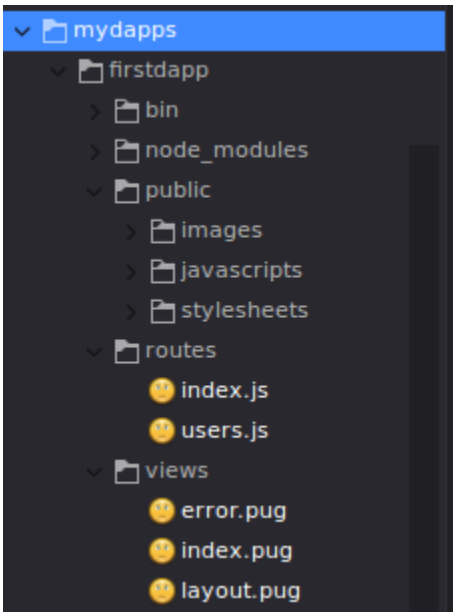
```
$ npm start
```

9. Webbrowser öffnen und http://localhost:3000 aufrufen.



Anpassen der ersten dApp

1. Starten Sie den **Komodo Editor** über das Startmenü unten links.
2. Navigieren Sie in das Verzeichnis ihrer Webseite.



3. Öffnen Sie die Datei **routes/index.js**
4. Anpassen des Titels der Webseite und Erstellen einer neuen Variable 'block':

routes/index.js

```
var express = require('express');
var router = express.Router();

/* GET home page. */
router.get('/', function(req, res, next) {

        var block = 'leer';
        res.render('index', { title: 'Hello World', block : block });

});

module.exports = router;
```

5. Öffnen Sie die Datei **views/index.pug** und passen sie die Darstellung der Webseite an**:**

views/index.pug (Einrückungen sind wichtig)

```
extends layout

block content
  h1= title
  div Aktueller Block: #{block}
```

Da serverseitiger Code geändert wurde (index.js), muss der Webserver wie folgt neu gestartet werden.

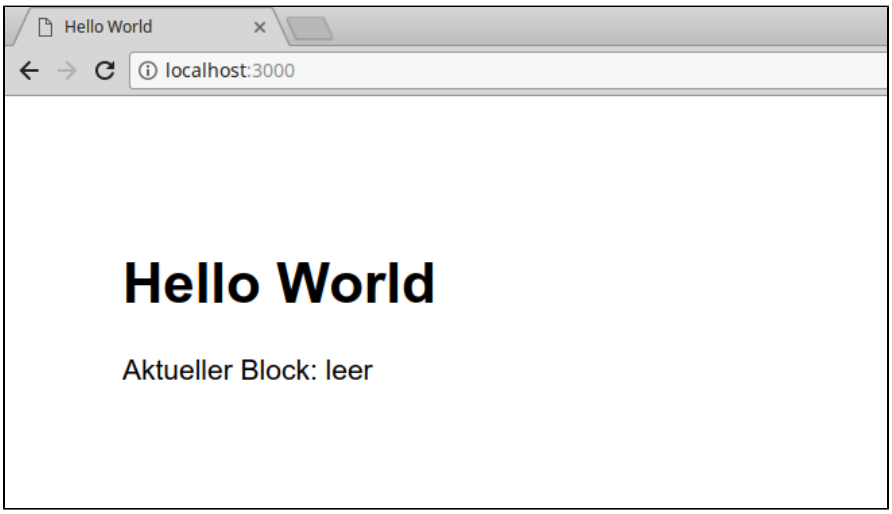
Drücken Sie dazu **Strg+C** in dem Terminalfenster, in dem Sie den Webserver gestartet haben.

Starten Sie den Webserver neu:

Terminalfenster / Konsole

```
$ npm start
```

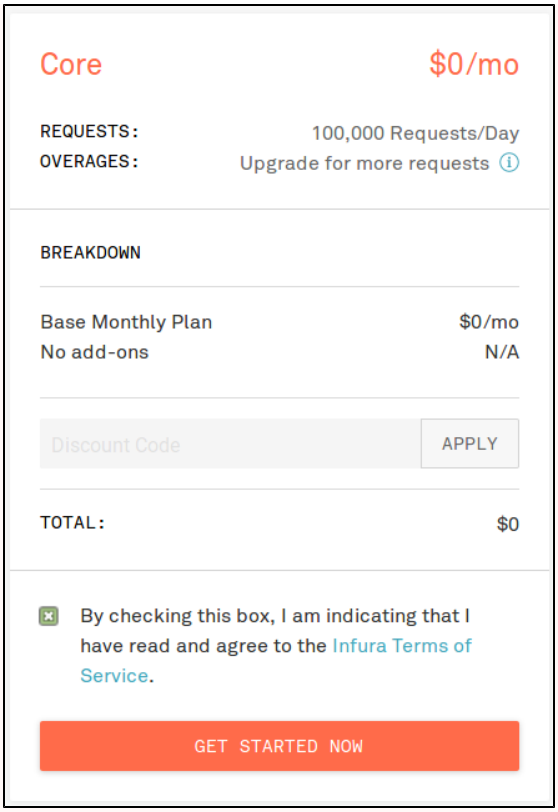
6. Aktualisieren Sie im Webbrowser die URL http://localhost:3000 mit der Taste **F5**.



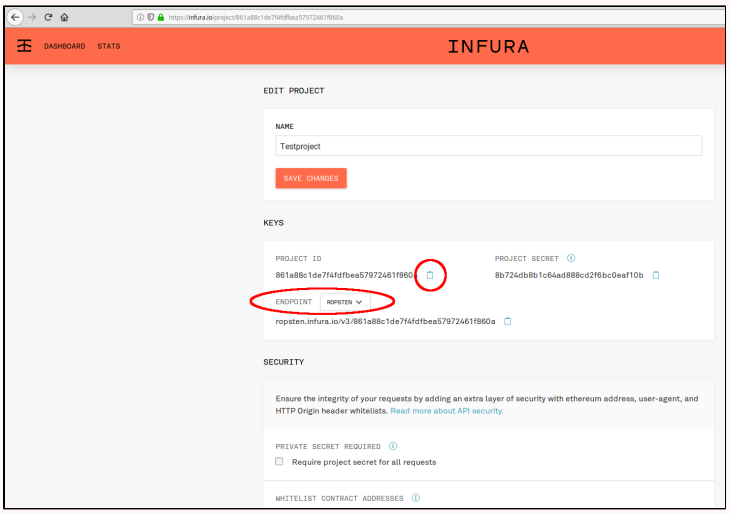
Serverseitige Verbindung mit der Ethereum Blockkette

Um auf die Blockchain zugreifen zu können, wollen wir https://infura.io verwenden. Infura ist ein Dienst, der eine Verbindung zu den Ethereum Netzwerken bereitstellt. Theoretisch können wir auch unseren eigenen Ethereum Node hosten. Dies ist aber recht ressourcenintensiv und würde den Rahmen der Praxissession sprengen.

- Legen Sie sich bitte einen neuen Infura Account an. Dazu benötigen Sie eine Email Adresse.
- Bestätigen Sie ihre Email Adresse und wählen Sie den kostenlosen "Core Plan" aus "**GET STARTED NOW**".



- Legen Sie bitte unter der Dashboard-Ansicht ein neues Projekt an: Klick auf "**CREATE NEW PROJECT**" und vergeben Sie einen Namen.
- Klicken Sie anschließend auf "**VIEW PROJECT**".
- Stellen Sie den **Endpoint auf Ropsten** und kopieren Sie sich die "**PROJECT ID**" welche als API Key dient:



1. Anpassen der Datei **routes/index.js** damit Sie sich die Webseite mit dem Ethereum Client verbindet und die aktuelle Blocknummer ausliest.

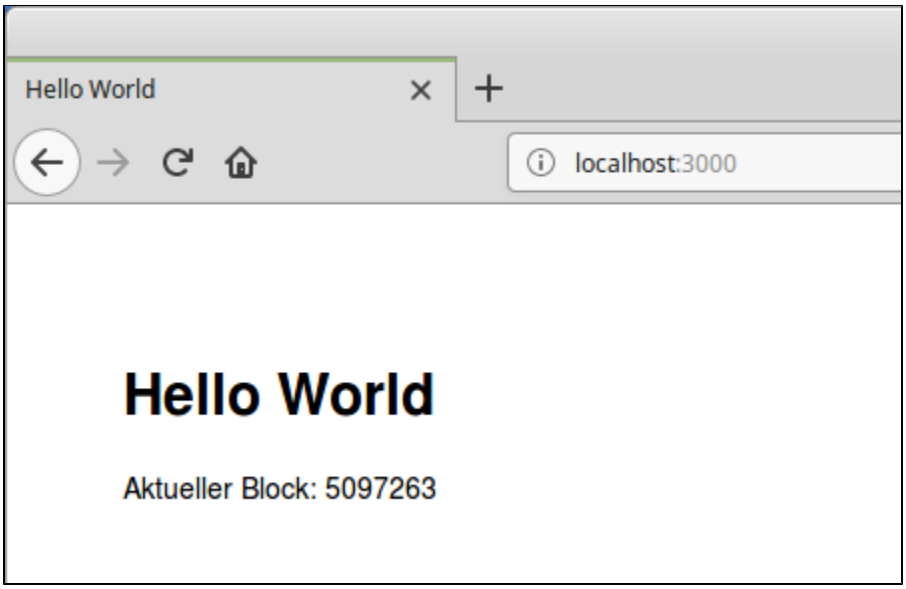
> Ersetzen Sie im folgenden Codebeispiel in Zeile 5 den Wert **<infura-api-key>** durch den von Ihnen oben kopierten Infura-Key. (Alternativer Key: 4e87j183Upn3DDvhx5M4)

routes/index.js

```
var express = require('express');
var router = express.Router();

var Web3JS = require("web3");
```

```
var web3 = new Web3JS(new Web3JS.providers.HttpProvider("https://ropsten.infura.io/<infura-api-key>"));

/* GET home page. */
router.get('/', function(req, res, next) {

  web3.eth.getBlockNumber().then(function(blocknumber){
    res.render('index', { title: 'Hello World', block : blocknumber });
  })
  .catch(function(error){
    res.render('index', { title: 'Hello World', block : 'konnte Block nicht laden'});
  });

});

module.exports = router;
```

2. **Starten Sie den Webserver neu** und rufen Sie im Webbrowser wieder http://localhost:3000 auf.



Clientseitige Verbindung mit der Ethereum Blockkette via Meta Mask

1. Installieren Sie das Browser Plugin **Meta Mask** falls Sie die Praxissession 2a - Ethereum Grundlagen verpasst haben sollten.
2. Legen Sie mit dem **Komodo Editor** im Ordner **public/javascripts** ihrer Webseite eine neue Datei **clientscript.js** an.
3. Öffnen Sie die Datei und fügen Sie folgenden Test-Code ein:

public/javascripts/clientscript.js

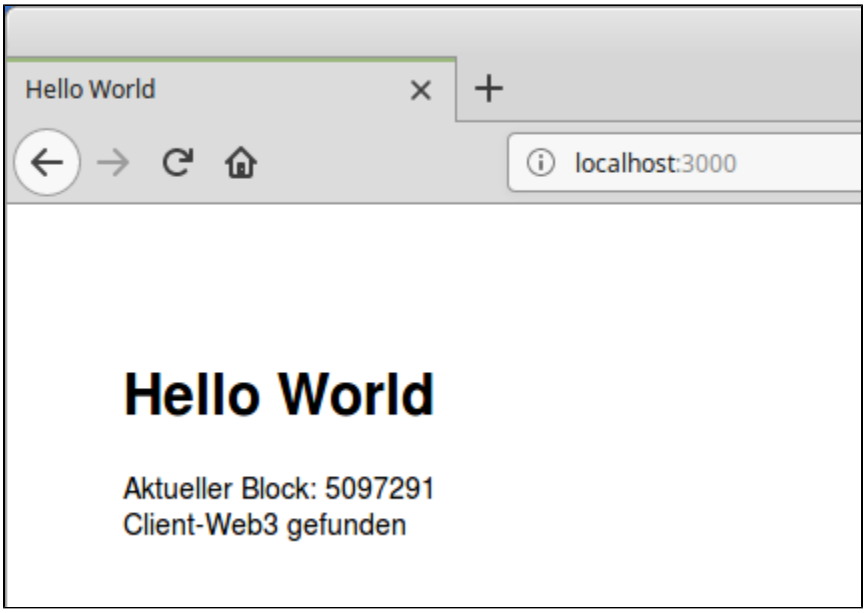
```
window.onload = function(e){

  var infoDiv = document.getElementById("info");

  if(typeof web3 !== 'undefined'){
    infoDiv.innerHTML = 'Client-Web3 gefunden';
  } else {
    infoDiv.innerHTML = 'Installieren Sie bitte <a href="https://metamask.io/">Meta Mask</a>.';
  }

}
```

4. Passen Sie anschließend die Datei **views/index.pug** wie folgt an:

views/index.pug

```
extends layout

block content

  h1= title
  div Aktueller Block: #{block}
  div(id='info')

  script(src="/javascripts/clientscript.js")
```

5. Aktualisieren Sie im Webbrowser die URL http://localhost:3000:



Einen Smart Contract auslesen und anzeigen

1. Um einen Smart Contact auslesen zu können, müssen wir eine Verbindung zur Blockkette herstellen. Dazu verwenden wir die Javascript Bibliothek **web3.**
2. Zunächst passen die Datei **routes/index.js** an.

Ersetzen Sie im folgenden Codebeispiel in Zeile 5 den Wert **<infura-api-key>** durch den von Ihnen oben erstellten Infura-Key.

routes/index.js

```
var express = require('express');
var router = express.Router();

var Web3JS = require("web3");
var web3 = new Web3JS(new Web3JS.providers.HttpProvider("https://ropsten.infura.io/<infura-api-key>"));

var abi = [
        {
                "constant": false,
                "inputs": [
                        {
                                "name": "_spender",
                                "type": "address"
                        },
                        {
                                "name": "_value",
                                "type": "uint256"
                        }
                ],
                "name": "approve",
                "outputs": [
```

```
                {
                        "name": "success",
                        "type": "bool"
                }
        ],
        "payable": false,
        "stateMutability": "nonpayable",
        "type": "function"
},
{
        "constant": false,
        "inputs": [
                {
                        "name": "_to",
                        "type": "address"
                },
                {
                        "name": "_value",
                        "type": "uint256"
                }
        ],
        "name": "transfer",
        "outputs": [
                {
                        "name": "success",
                        "type": "bool"
                }
        ],
        "payable": false,
        "stateMutability": "nonpayable",
        "type": "function"
},
{
        "constant": false,
        "inputs": [
                {
                        "name": "_from",
                        "type": "address"
                },
                {
                        "name": "_to",
                        "type": "address"
                },
                {
                        "name": "_value",
                        "type": "uint256"
                }
        ],
        "name": "transferFrom",
        "outputs": [
                {
                        "name": "success",
                        "type": "bool"
                }
        ],
        "payable": false,
        "stateMutability": "nonpayable",
        "type": "function"
},
{
        "inputs": [
                {
                        "name": "_initialSupply",
                        "type": "uint256"
                },
                {
                        "name": "_name",
                        "type": "string"
                },
                {
                        "name": "_symbol",
```

```
                        "type": "string"
                }
        ],
        "payable": false,
        "stateMutability": "nonpayable",
        "type": "constructor"
},
{
        "anonymous": false,
        "inputs": [
                {
                        "indexed": true,
                        "name": "_from",
                        "type": "address"
                },
                {
                        "indexed": true,
                        "name": "_to",
                        "type": "address"
                },
                {
                        "indexed": false,
                        "name": "_value",
                        "type": "uint256"
                }
        ],
        "name": "Transfer",
        "type": "event"
},
{
        "anonymous": false,
        "inputs": [
                {
                        "indexed": true,
                        "name": "_owner",
                        "type": "address"
                },
                {
                        "indexed": true,
                        "name": "_spender",
                        "type": "address"
                },
                {
                        "indexed": false,
                        "name": "_value",
                        "type": "uint256"
                }
        ],
        "name": "Approval",
        "type": "event"
},
{
        "constant": true,
        "inputs": [
                {
                        "name": "",
                        "type": "address"
                },
                {
                        "name": "",
                        "type": "address"
                }
        ],
        "name": "allowance",
        "outputs": [
                {
                        "name": "",
                        "type": "uint256"
                }
        ],
        "payable": false,
```

```
                "stateMutability": "view",
                "type": "function"
        },
        {
                "constant": true,
                "inputs": [
                        {
                                "name": "",
                                "type": "address"
                        }
                ],
                "name": "balanceOf",
                "outputs": [
                        {
                                "name": "",
                                "type": "uint256"
                        }
                ],
                "payable": false,
                "stateMutability": "view",
                "type": "function"
        },
        {
                "constant": true,
                "inputs": [],
                "name": "name",
                "outputs": [
                        {
                                "name": "",
                                "type": "string"
                        }
                ],
                "payable": false,
                "stateMutability": "view",
                "type": "function"
        },
        {
                "constant": true,
                "inputs": [],
                "name": "symbol",
                "outputs": [
                        {
                                "name": "",
                                "type": "string"
                        }
                ],
                "payable": false,
                "stateMutability": "view",
                "type": "function"
        },
        {
                "constant": true,
                "inputs": [],
                "name": "totalSupply",
                "outputs": [
                        {
                                "name": "",
                                "type": "uint256"
                        }
                ],
                "payable": false,
                "stateMutability": "view",
                "type": "function"
        }
];

var tokenAdress = '0x1225446678163b14859eed119831a68576fc15ae';
var token = new web3.eth.Contract(abi, tokenAdress);

/* GET home page. */
router.get('/', function(req, res, next) {
```

```
  var blk;
  var tkn;

  web3.eth.getBlockNumber().then(function(blocknumber){
    blk = blocknumber;
    return token.methods.name().call();
  })
  .then(function(name){
    tkn = name;
  })
  .then(function(){
    res.render('index', { title: 'Hello World', block : blk, tokenname : tkn });
  })
  .catch(function(error){
    console.log(error);
    res.render('index', { title: 'Hello World', block : 'konnte Block nicht laden'});
  });

});

module.exports = router;
```

3. Nun erweitern wir wieder die Anzeige um den Namen des Tokens, den wir aus der Blockkette gelesen haben.

views/index.pug

```
extends layout

block content

  h1= title
  div Aktueller Block: #{block}
  div Token Name: #{tokenname}
  br
  div(id='info')

  script(src="/javascripts/clientscript.js")
```

4. Da wir wieder serverseitigen Code geändert haben, muss der Webserver wieder neu gestartet werden.

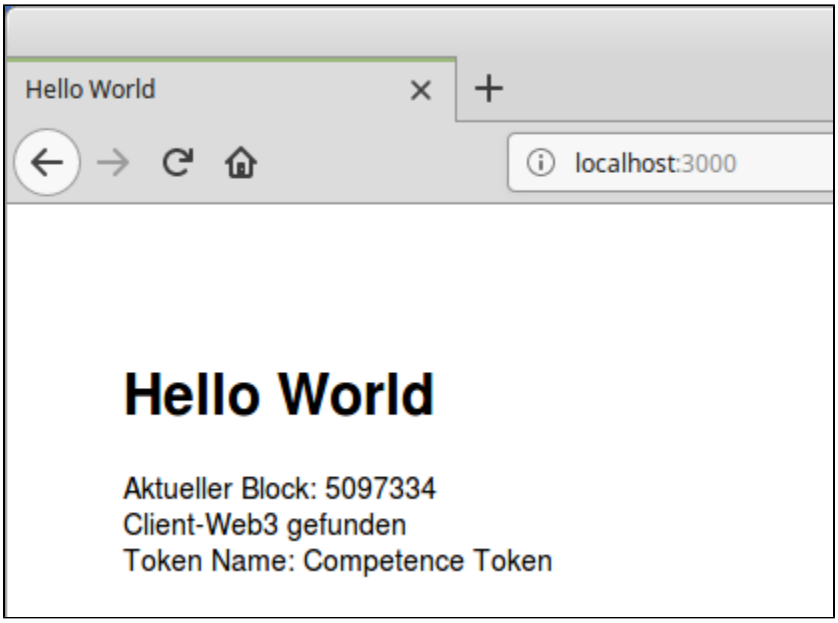
   Drücken Sie dazu **Strg+C** in dem Terminalfenster, in dem Sie den Webserver gestartet haben.

   Starten Sie den Webserver neu:

   Terminalfenster / Konsole

   ```
   $ npm start
   ```

5. Aktualisieren Sie im Webbrowser die URL http://localhost:3000:



Zusatz:

Als Zusatzaufgabe sollen Sie sich im Browser ihre Anzahl Competence Token anzeigen lassen. Das Besondere ist, dass dies vollkommen Client-seitig mittels des MetaMask Plugins geschehen soll.

Meta Mask mit der Dapp verknüpfen.

public/javascripts/clientscript.js

```
window.onload = function(e){

  var infoDiv = document.getElementById("info");
  var account;

  // check if metamask is available
  if (typeof window.ethereum !== 'undefined') {

    // load web3 client library from metamask
    web3 = new Web3(ethereum);
    console.log("Web3 Api Version: "+web3.version.api);

    // enable metamask and load account
    ethereum.enable().then((a) => {
      account = a[0];
      infoDiv.innerHTML = ''+ account +' mit Dapp verknüpft';
    });

  } else {
    infoDiv.innerHTML = 'Installieren Sie bitte <a href="https://metamask.io/">Meta Mask</a>.';
  }

}
```

Competence Token Contract Client-seitig laden und Anzahl Token des verbundenen Accounts auslesen

public/javascripts/clientscript.js

```
window.onload = function(e){

  var abi = [
      {
          "constant": false,
          "inputs": [
              {
                  "name": "_spender",
                  "type": "address"
              },
              {
                  "name": "_value",
                  "type": "uint256"
              }
          ],
          "name": "approve",
          "outputs": [
              {
                  "name": "success",
                  "type": "bool"
              }
          ],
          "payable": false,
          "stateMutability": "nonpayable",
          "type": "function"
      },
      {
          "constant": false,
          "inputs": [
              {
                  "name": "_to",
                  "type": "address"
              },
              {
                  "name": "_value",
                  "type": "uint256"
              }
          ],
          "name": "transfer",
          "outputs": [
              {
                  "name": "success",
                  "type": "bool"
              }
          ],
          "payable": false,
          "stateMutability": "nonpayable",
          "type": "function"
      },
      {
          "constant": false,
          "inputs": [
              {
                  "name": "_from",
                  "type": "address"
              },
              {
                  "name": "_to",
                  "type": "address"
              },
              {
                  "name": "_value",
                  "type": "uint256"
              }
          ],
          "name": "transferFrom",
          "outputs": [
              {
                  "name": "success",
                  "type": "bool"
```

```
            }
        ],
        "payable": false,
        "stateMutability": "nonpayable",
        "type": "function"
    },
    {
        "inputs": [
            {
                "name": "_initialSupply",
                "type": "uint256"
            },
            {
                "name": "_name",
                "type": "string"
            },
            {
                "name": "_symbol",
                "type": "string"
            }
        ],
        "payable": false,
        "stateMutability": "nonpayable",
        "type": "constructor"
    },
    {
        "anonymous": false,
        "inputs": [
            {
                "indexed": true,
                "name": "_from",
                "type": "address"
            },
            {
                "indexed": true,
                "name": "_to",
                "type": "address"
            },
            {
                "indexed": false,
                "name": "_value",
                "type": "uint256"
            }
        ],
        "name": "Transfer",
        "type": "event"
    },
    {
        "anonymous": false,
        "inputs": [
            {
                "indexed": true,
                "name": "_owner",
                "type": "address"
            },
            {
                "indexed": true,
                "name": "_spender",
                "type": "address"
            },
            {
                "indexed": false,
                "name": "_value",
                "type": "uint256"
            }
        ],
        "name": "Approval",
        "type": "event"
    },
    {
        "constant": true,
```

```
        "inputs": [
            {
                "name": "",
                "type": "address"
            },
            {
                "name": "",
                "type": "address"
            }
        ],
        "name": "allowance",
        "outputs": [
            {
                "name": "",
                "type": "uint256"
            }
        ],
        "payable": false,
        "stateMutability": "view",
        "type": "function"
    },
    {
        "constant": true,
        "inputs": [
            {
                "name": "",
                "type": "address"
            }
        ],
        "name": "balanceOf",
        "outputs": [
            {
                "name": "",
                "type": "uint256"
            }
        ],
        "payable": false,
        "stateMutability": "view",
        "type": "function"
    },
    {
        "constant": true,
        "inputs": [],
        "name": "name",
        "outputs": [
            {
                "name": "",
                "type": "string"
            }
        ],
        "payable": false,
        "stateMutability": "view",
        "type": "function"
    },
    {
        "constant": true,
        "inputs": [],
        "name": "symbol",
        "outputs": [
            {
                "name": "",
                "type": "string"
            }
        ],
        "payable": false,
        "stateMutability": "view",
        "type": "function"
    },
    {
        "constant": true,
        "inputs": [],
```

```
        "name": "totalSupply",
        "outputs": [
            {
                "name": "",
                "type": "uint256"
            }
        ],
        "payable": false,
        "stateMutability": "view",
        "type": "function"
    }
];

var tokenAdress = '0x1225446678163b14859eed119831a68576fc15ae'
var infoDiv = document.getElementById("info");
var account;

// check if metamask is available
if (typeof window.ethereum !== 'undefined') {

  // load web3 client library from metamask
  web3 = new Web3(ethereum);
  console.log("Web3 Api Version: "+web3.version.api);

  // enable metamask and load account
  ethereum.enable().then((a) => {
    account = a[0];
    infoDiv.innerHTML = ''+ account +' mit Dapp verknüpft';

    // load token contract
    return loadContract(tokenAdress, abi);
  }).then((tokencontract) => {

    // call method blanceOf from token contract
    tokencontract.balanceOf(account,(e,balance)=>{
      console.log(balance.toString());
      infoDiv.innerHTML += '<br>Guthaben: '+ balance.toString() +' Competence Token';
    });

  });

} else {
  infoDiv.innerHTML = 'Installieren Sie bitte <a href="https://metamask.io/">Meta Mask</a>.';
}

function loadContract(address, abi){
  var Contract = web3.eth.contract(abi);
  var instance = Contract.at(address);
  return new Promise((resolve, reject) => { resolve(instance) });
}

}
```