



Create A Simple Dapp

Tip

We will be building this [app](#)🔗

Project Setup

Before you set up make sure you've visited and gone through our [Getting Started Guide](#)

Make sure you have:

1. The [MetaMask Extension](#)🔗 downloaded.
2. Node.js [Downloaded and Installed](#)🔗
3. Clone/Download the [Project Files](#)🔗 from GitHub.
4. Your favorite Text Editor or IDE installed. I personally like [Visual Studio Code](#)🔗

Open Project Folder

Open the project folder. Navigate to `start -> index.html` , and look at the comment stating part 1. This is the UI for the Simple Dapp with all of the layout and buttons that will help us learn the basics of connecting to our MetaMask extension. We will be using/building off of this entire section for the first part of the tutorial.

Install Dependencies

Open a terminal and make sure your terminal is inside the base directory of the `start/` folder. Inside the folder, the files should look like this:

```
.  
├─ index.html  
├─ contract.js
```



You'll have some more files but that's nothing to worry about!

Open your terminal and navigate into the start folder. In this folder run:

```
yarn install
```

sh

This will install all the necessary dependencies we'll need for our project. This will have created a `node_modules/` folder where all the dependencies are stored.

Next run:

```
yarn run serve
```

sh

Navigate to `http://localhost:9011`

Basic Action(Part 1)

Now let's navigate into the `contract.js` file inside your start folder.

Your file should look something like this. Don't worry about lines 1-31.

```
const forwarderOrigin = 'http://localhost:9010';

const initialize = () => {
  //You will start here
};

window.addEventListener('DOMContentLoaded', initialize);
```

js

As soon as the content in the DOM is loaded we are calling our initialize function. Now before we start writing any code let's look at our task list for the first part of this app.

What we'll cover in part one:



- Display our network number
- Display our ChainId
- Display our Accounts

Connecting to the MetaMask Wallet

The first thing we need to do in our Dapp is to connect to our MetaMask Wallet.

1. We need to create a function to see if the MetaMask Chrome extension is installed
2. If MetaMask is not installed we:
 1. Change our `connectButton` to `Click here to install MetaMask`
 2. When clicking that button it should take us to a page that will allow us to install the extension
 3. Disable the button
3. If MetaMask is installed we:
 1. Change our `connectButton` to `Connect`
 2. When clicking that button it should allow us to connect to our MetaMask wallet
 3. Disable the button

Let's get to it!!

MetaMask Extension Check

In our code we need to connect to our button from our index.html

```
const initialize = () => {  
  //Basic Actions Section  
  const onboardButton = document.getElementById('connectButton');  
};
```

js

Next we create a check function called `isMetaMaskInstalled` to see if the MetaMask extension is installed



```
const onboardButton = document.getElementById('connectButton');

//Created check function to see if the MetaMask extension is installed
const isMetaMaskInstalled = () => {
  //Have to check the ethereum binding on the window object to see if it's installed
  const { ethereum } = window;
  return Boolean(ethereum && ethereum.isMetaMask);
};
```

Next we need to create a `MetaMaskClientCheck` function to see if we need to change the button text based on if the MetaMask Extension is installed or not.

```
const initialize = () => {
  //Basic Actions Section
  const onboardButton = document.getElementById('connectButton');

  //Created check function to see if the MetaMask extension is installed
  const isMetaMaskInstalled = () => {
    //Have to check the ethereum binding on the window object to see if it's installed
    const { ethereum } = window;
    return Boolean(ethereum && ethereum.isMetaMask);
  };

  //-----Inserted Code-----\\
  const MetaMaskClientCheck = () => {
    //Now we check to see if MetaMask is installed
    if (!isMetaMaskInstalled()) {
      //If it isn't installed we ask the user to click to install it
      onboardButton.innerText = 'Click here to install MetaMask!';
    } else {
      //If it is installed we change our button text
      onboardButton.innerText = 'Connect';
    }
  };
  MetaMaskClientCheck();
  //-----/Inserted Code-----\\
};
```



MetaMask "Not Installed" Dapp Flow


In our code block where MetaMask isn't installed and we ask the user to `'Click here to install MetaMask!'`, we need to ensure that if our button is clicked we:

1. Redirect the user to the proper page to install the extension
2. Disable the button

```
const MetaMaskClientCheck = () => {  
  //Now we check to see if Metmask is installed  
  if (!isMetaMaskInstalled()) {  
    //If it isn't installed we ask the user to click to install it  
    onboardButton.innerText = 'Click here to install MetaMask!';  
    //When the button is clicked we call this function  
    onboardButton.onclick = onClickInstall;  
    //The button is now disabled  
    onboardButton.disabled = false;  
  } else {  
    //If it is installed we change our button text  
    onboardButton.innerText = 'Connect';  
  }  
};  
MetaMaskClientCheck();
```

We've created a function that will be called when we click the button and disable it. Let's dive into the `onClickInstall` function and create the logic inside of it.

Tip

For this part we will be using the '@metamask/onboarding' library we installed during the npm install. To learn more visit [here](#) 

Inside this function we want to:

1. Change the text of the button to `Onboarding in progress`
2. Disable the button



Above your `MetaMaskClientCheck` function write/insert this code.

```
js
//We create a new MetaMask onboarding object to use in our app
const onboarding = new MetaMaskOnboarding({ forwarderOrigin });

//This will start the onboarding process
const onClickInstall = () => {
  onboardButton.innerText = 'Onboarding in progress';
  onboardButton.disabled = true;
  //On this object we have startOnboarding which will start the onboarding process for
  onboarding.startOnboarding();
};
```

GREAT! Now if our end user doesn't have the MetaMask Extension they can install it. When they refresh the page the ethereum window object will be there and we can get on to connecting their MetaMask wallet to our Dapp!

MetaMask "Installed" Dapp Flow

Next we need to revisit our `MetaMaskClientCheck` function and create similar functionality that we did in our "MetaMask Not Installed" block in our "MetaMask Is Installed" block of code.

```
js
const MetaMaskClientCheck = () => {
  //Now we check to see if Metmask is installed
  if (!isMetaMaskInstalled()) {
    //If it isn't installed we ask the user to click to install it
    onboardButton.innerText = 'Click here to install MetaMask!';
    //When the button is clicked we call th is function
    onboardButton.onclick = onClickInstall;
    //The button is now disabled
    onboardButton.disabled = true;
  } else {
    //If MetaMask is installed we ask the user to connect to their wallet
    onboardButton.innerText = 'Connect';
    //When the button is clicked we call this function to connect the users MetaMask Wa
    onboardButton.onclick = onClickConnect;
    //The button is now disabled
```



```
MetaMaskClientCheck();
```

Now we've created a function that will be called whenever we click the button to trigger a connection to our wallet, disabling the button. Next, let's dive into the `onClickConnect` function and build the logic we need inside of it.

Inside this function we want to:

1. Create an async function that will try to call the 'eth_requestAccounts' RPC method
2. Catch any errors and log them to the console

Under your `onClickInstall` function write/insert this code.

```
const onClickConnect = async () => {
  try {
    // Will open the MetaMask UI
    // You should disable this button while the request is pending!
    await ethereum.request({ method: 'eth_requestAccounts' });
  } catch (error) {
    console.error(error);
  }
};
```

js

Great! Now once you click the button the MetaMask Extension will pop up and connect your wallet.

Get Ethereum Accounts

After this what we'd like to do next is whenever we press the `eth_accounts` button we'd like to get our Ethereum account and display it. Replace the existing `const onboardButton` at the top of the `initialize()` function with the following three buttons:

```
//Basic Actions Section
const onboardButton = document.getElementById('connectButton');
```

js



Now that we've grabbed our `eth_accounts` button and our paragraph field to display it in we now have to grab the data.

Under our `MetaMaskClientCheck()` function let's write/insert the code below.

```
//Eth_Accounts-getAccountsButton
getAccountsButton.addEventListener('click', async () => {
  //we use eth_accounts because it returns a list of addresses owned by us.
  const accounts = await ethereum.request({ method: 'eth_accounts' });
  //We take the first address in the array of addresses and display it
  getAccountsResult.innerHTML = accounts[0] || 'Not able to get accounts';
});
```

js

If you have already connected to your wallet in the previous section of the tutorial, you can go into MetaMask's "Connected Sites" menu and remove the localhost connection. This will enable us to test both buttons again. If we refresh our page, and click the "ETH_ACCOUNTS" button, we should see `'eth_accounts result: Not able to get accounts'` .

Let's go ahead and press the "Connect" button again, and confirm the "Connect With MetaMask" prompt and "Connect". Now we can click the "ETH_ACCOUNTS" button again and we should see our MetaMask account public address.

CONGRATULATIONS!

We have just completed building out our Basic Actions functionality. You know how to Connect to MetaMask, see your connected apps and remove them, as well as retrieve accounts.

Now on to our next step, showing our statuses.

Preview of the completed code up until this point of the tutorial:

```
const forwarderOrigin = 'http://localhost:9010';

const initialize = () => {
  //Basic Actions Section
  const onboardButton = document.getElementById('connectButton');
```

js



```
//Created check function to see if the MetaMask extension is installed
const isMetaMaskInstalled = () => {
  //Have to check the ethereum binding on the window object to see if it's installed
  const { ethereum } = window;
  return Boolean(ethereum && ethereum.isMetaMask);
};

//We create a new MetaMask onboarding object to use in our app
const onboarding = new MetaMaskOnboarding({ forwarderOrigin });

//This will start the onboarding process
const onClickInstall = () => {
  onboardButton.innerText = 'Onboarding in progress';
  onboardButton.disabled = true;
  //On this object we have startOnboarding which will start the onboarding process for
  onboarding.startOnboarding();
};

const onClickConnect = async () => {
  try {
    // Will open the MetaMask UI
    // You should disable this button while the request is pending!
    await ethereum.request({ method: 'eth_requestAccounts' });
  } catch (error) {
    console.error(error);
  }
};

const MetaMaskClientCheck = () => {
  //Now we check to see if Metmask is installed
  if (!isMetaMaskInstalled()) {
    //If it isn't installed we ask the user to click to install it
    onboardButton.innerText = 'Click here to install MetaMask!';
    //When the button is clicked we call this function
    onboardButton.onclick = onClickInstall;
    //The button is now disabled
    onboardButton.disabled = false;
  } else {
    //If MetaMask is installed we ask the user to connect to their wallet
    onboardButton.innerText = 'Connect';
  }
}
```



```
// the button is now disabled
onboardButton.disabled = false;
}
};

//Eth_Accounts-getAccountsButton
getAccountsButton.addEventListener('click', async () => {
  //we use eth_accounts because it returns a list of addresses owned by us.
  const accounts = await ethereum.request({ method: 'eth_accounts' });
  //We take the first address in the array of addresses and display it
  getAccountsResult.innerHTML = accounts[0] || 'Not able to get accounts';
});

MetaMaskClientCheck();
};

window.addEventListener('DOMContentLoaded', initialize);
```

[Edit this page on GitHub](#) 

Last Updated: 4/7/2022, 4:12:26 PM

[← Patching Dependencies](#)

[Contributors →](#)