

PROJETO 1 - Preparação para o Laboratório de Smart Contracts e Criptomoedas

Curso Blockchain Developer

Material produzido por [bbchain](#).

Os materiais publicados nesta página são protegidos por direitos autorais e são de propriedade da bbchain, juntamente com quaisquer outros direitos de propriedade intelectual sobre tais materiais. Todos os direitos reservados. Nenhuma parte desta página pode ser copiada, reproduzida, apresentada em público, transmitida, carregada, divulgada, distribuída, modificada ou tratada de nenhuma maneira sem o consentimento prévio por escrito da bbchain e, mesmo com tal consentimento, a fonte e os direitos de propriedade devem ser reconhecidos.

1. Objetivo

Instalar, configurar e testar as ferramentas que compõem um ambiente de desenvolvimento de Smart Contracts em Solidity e de Decentralized Application (dApps) para Ethereum. Esse ambiente será utilizado nos demais projetos deste laboratório.

2. Pré-Requisitos

Computador com sistema operacional Windows, Linux ou MacOS e acesso a Internet.

3. Materiais (Checklist)

- [Notebook com Sistema Operacional Windows, Linux ou MacOS]
- [Google Chrome] (<https://www.google.com/chrome/>)
- [Git] (<https://git-scm.com/downloads>)
- [Node.js] (<https://nodejs.org/en/>)
- [MetaMask] (<https://metamask.io/>)
- [Remix] (<http://remix.ethereum.org>)
- [remixd] (<https://www.npmjs.com/package/remixd>)
- [OpenZeppelin] (<https://openzeppelin.org/>)
- [Windows Build Tools (*apenas para Windows)] (<https://www.npmjs.com/package/windows-build-tools>)
- [GETH] (<https://geth.ethereum.org/install>)
- [serve] (<https://www.npmjs.com/package/serve>)
- [Visual Studio Code] (<https://code.visualstudio.com/download>) **(ou outra IDE de sua preferência)**
- [Terminal de Comando]

4. Ferramentas

I. Google Chrome

O navegador [Google Chrome](#) será utilizado para usar navegar em ferramentas web e instalar o plugin MetaMask (ver a seguir).

II. Git

O [Git](#) é um sistema de controle de versão. Ele é necessário para a instalação e de alguns módulos do Node.js (ver a seguir).

III. Node.js

O [Node.js](#) é um ambiente execução de Javascript no lado do servidor.

Por padrão, a instalação do Node.js (`node`) também instala o `npm`, que é um gerenciador de módulos do `node`.

Vamos usar o `npm` para instalar os módulos `remixd` e `serve` (ver a seguir) que foram desenvolvidos para rodar no `node`.

IV. MetaMask

O [MetaMask](#) é um plugin que permite interagir com dApps Ethereum diretamente do navegador, sem precisar rodar um full node. Além disso permite o gerenciamento e armazenamento de chaves privadas e a assinatura mais segura de transações.

V. Remix

O [Remix](#) é um *Integrated Development Environment* (IDE) Web para o desenvolvimento de Smart Contracts e interação com a blockchain do Ethereum. O Remix inclui ferramentas para debug de transações, compilação de código Solidity e interação ativa com Smart Contracts em redes Ethereum públicas e privadas ou uma rede virtual simulada em Javascript.

O desenvolvimento e os testes dos Smart Contracts propostos nos projetos serão realizadas utilizando esta ferramenta.

VI. remixd

O [remixd](#) é um módulo `node` que permite que o Remix consiga importar arquivos disponíveis no computador.

Iremos utilizar o `remixd` para utilizar os Smart Contracts do OpenZeppelin no Remix.

VII. OpenZeppelin

O [OpenZeppelin](#) é um framework que implementa Smart Contracts Solidity reutilizáveis segundo as melhores práticas de desenvolvimento e segurança para Ethereum.

Iremos utilizar o OpenZeppelin para reduzir o risco de vulnerabilidades e aumentar a produtividade, por meio da reutilização de Smart Contracts padronizados, como ERC20 e ERC721.

VIII. Windows Build Tools (*apenas para Windows)

Para usuários Windows, é necessário instalar [Windows Build Tools](#) para que o `npm` consiga instalar alguns módulos do Node.js.

O Windows Build Tools instala e configura o Visual C++ Build Tools e o Python 2.7.



[Windows Vista / 7] exigem a instalação adicional do .NET Framework 4.5.1 separadamente.

IX. Web3.js

O [Web3.js](#) é uma biblioteca Javascript que permite desenvolver aplicações web que interajam com Smart Contracts em redes Ethereum.

X. GETH

O [GETH](#) é um cliente Ethereum implementado em Go que permite rodar um full nodes Ethereum (para minerar) ou criar uma blockchain em uma rede privada Ethereum.

Vamos utilizar o GETH para criar uma nova instância de blockchain Ethereum localmente.

XI. serve

O [serve](#) é um módulo do `node` utilizado para criar um servidor HTTP a partir de um diretório.

Iremos utilizar o `serve` para rodar uma aplicação web para interagir com Smart Contracts de uma rede Ethereum.

XII. Visual Studio Code

O [Visual Studio Code](#) é uma IDE que podemos utilizar para escrever a aplicação web com Javascript e `web3.js`.

Caso esteja familiarizado com outra IDE, não é necessário usar esta ferramenta.

5. Preparação

Para o desenvolvimento dos projetos deste laboratório, serão necessárias uma série de preparações.

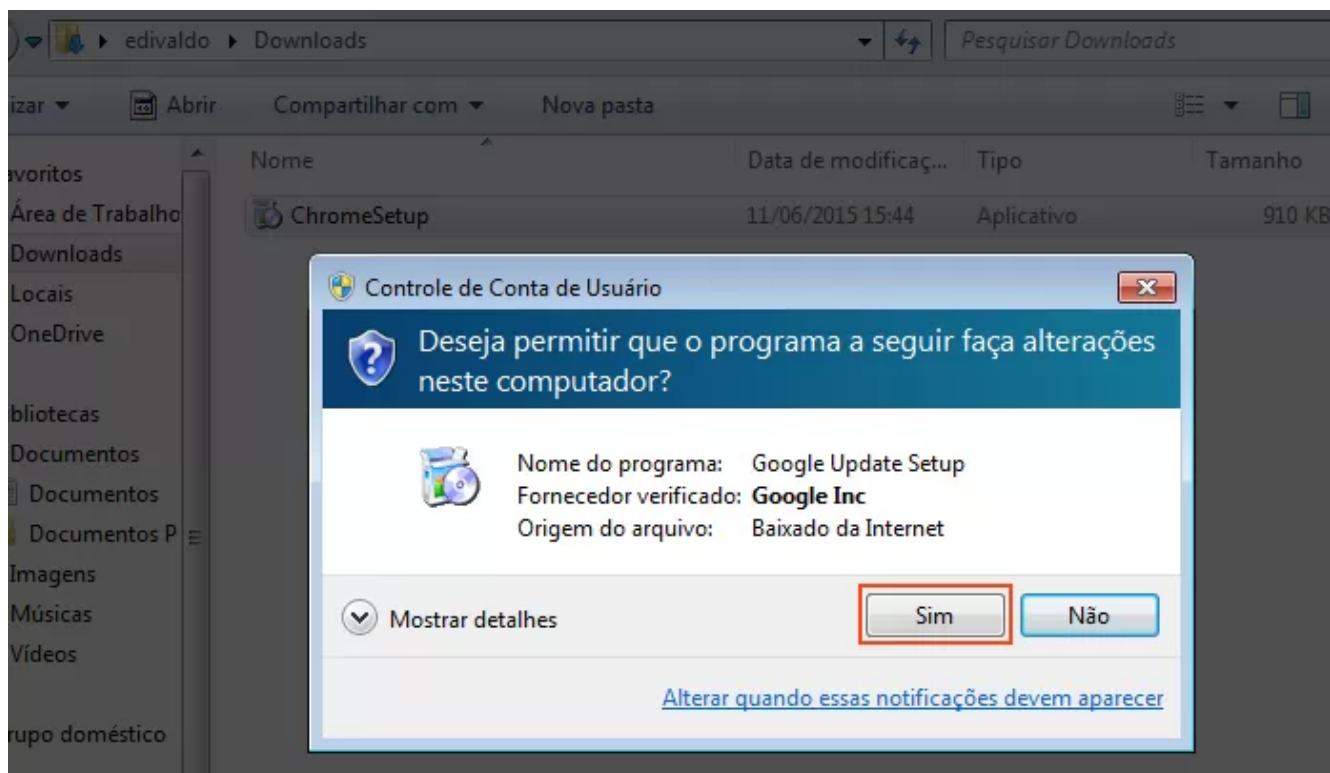
Como os procedimentos podem variar de acordo com o sistema operacional, vamos separar as instruções em três sistemas operacionais: Windows, MacOS e Linux (Ubuntu).

Windows

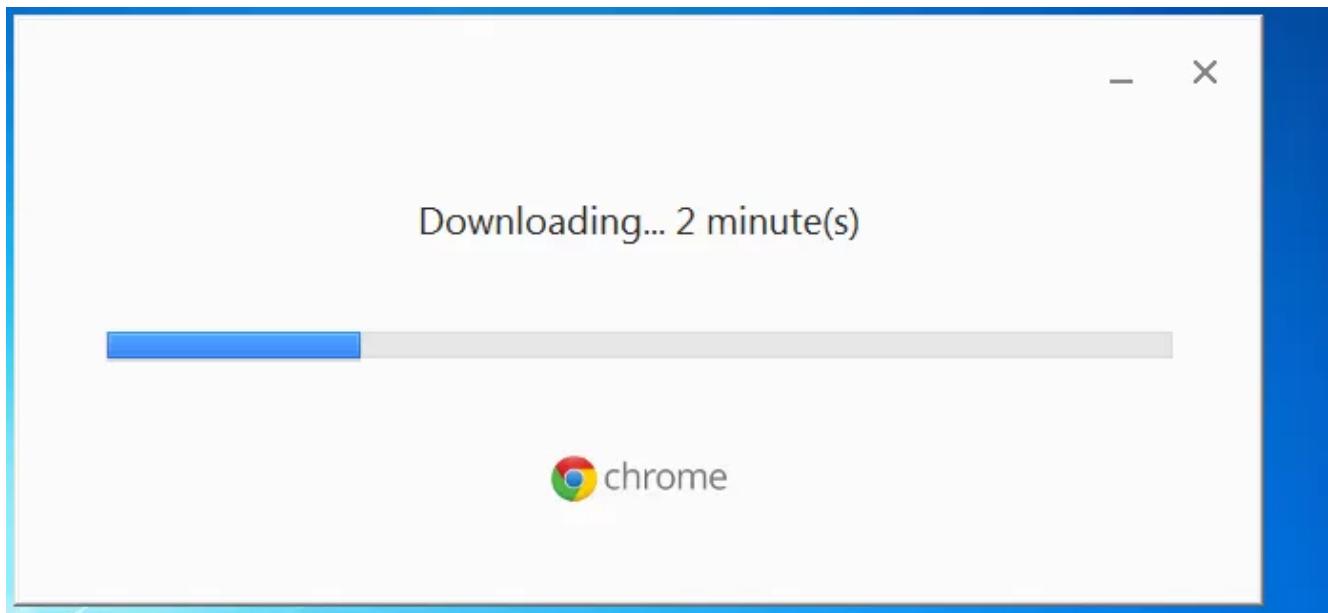
I. Google Chrome

Instalação do Google Chrome

1. Acesse o site do Google Chrome: <https://www.google.com/chrome>
2. Clique em **Fazer o download do Google Chrome**
3. Leia os termos de Serviço e clique em **Aceitar e Instalar**
4. Abra o arquivo `ChromeSetup.exe` baixado
5. Aceite a instalação em modo de desenvolvedor



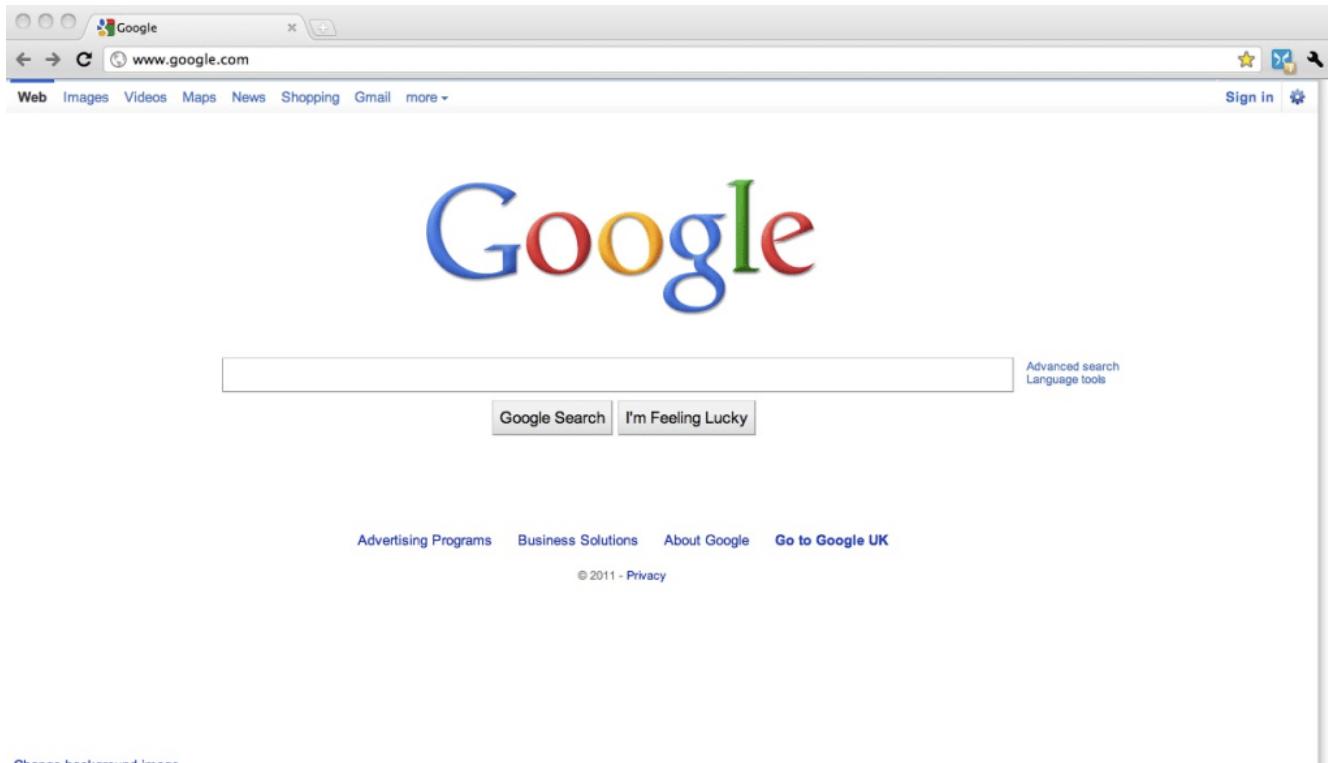
6. Aguarde a finalização do instalador



Teste do Google Chrome

1. Pressione as teclas Windows
2. Digite Google Chrome, localize o ícone  e pressione ENTER

Resultado:



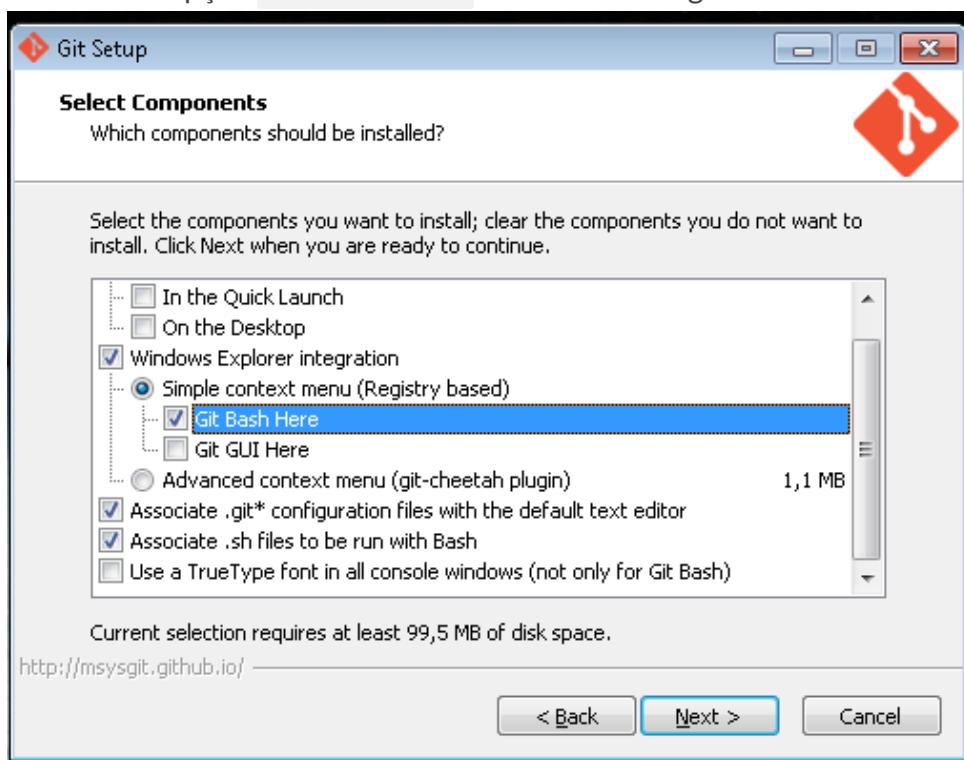
II. Git

Instalação do Git

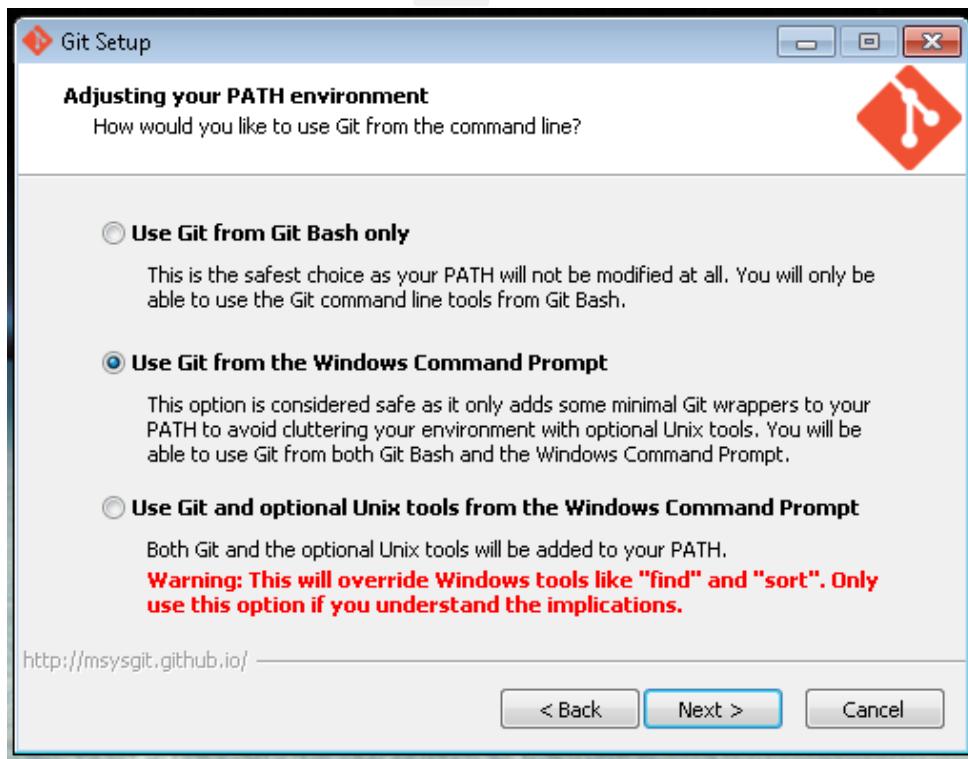
- Acesse a página de instalação do Git: <https://git-scm.com/download/win>
- Clique em `click here to download manually` se o download não começar em alguns instantes
- Abra o arquivo `Git-2.x.x-64-bit.exe`



- Pressione duas vezes o botão `Next`
- Selecione a opção `Git Bash Here` conforme a imagem abaixo

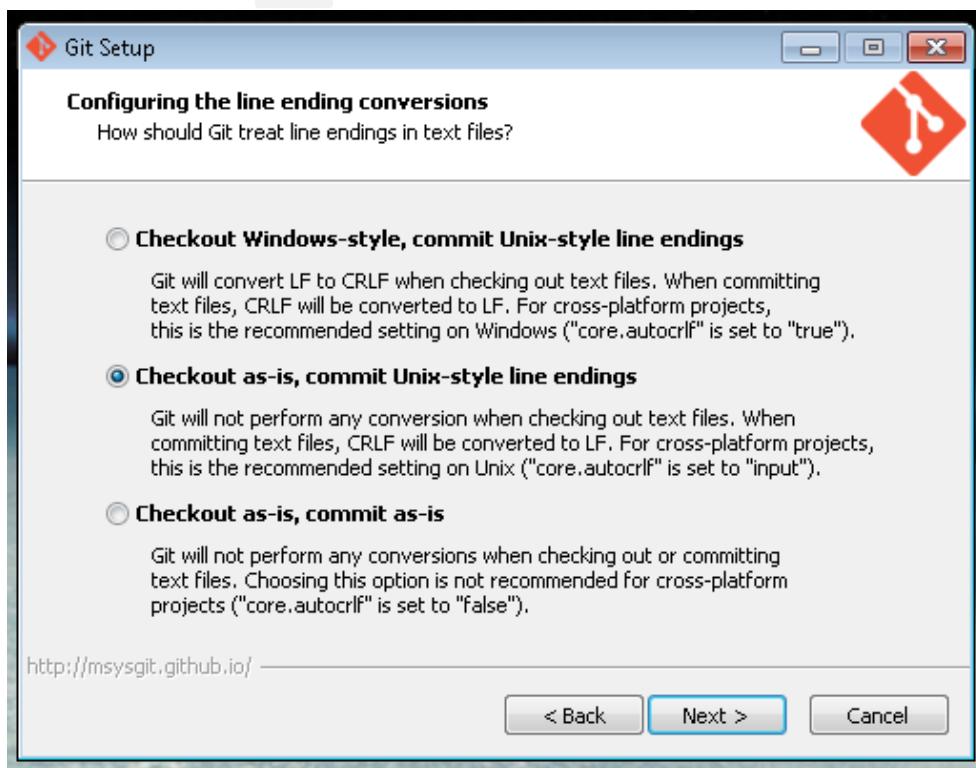


6. Pressione novamente o botão `Next` duas vezes



7. Selecione a opção `Use Git from the Windows Command Prompt`

8. Pressione o botão `Next` até visualizar a tela abaixo



Teste do Git

- Abra o prompt de comando do Windows utilizando o comando `windows+r` digitando `cmd` e pressionando `enter`
- Digite o comando abaixo:

```
git --version
```

Resultado:

```
git version 2.x.x
```

III. Node.js

1. Acesse a página de instalação do Node.js: <https://nodejs.org/en/>
2. Selecione a versão Current - Lastes Features
3. Abra o arquivo node-v10.x.x.msi
4. Clique em Continuar duas vezes e Instalar
5. Aguarde o fim da instalação

Teste do Node.js

1. Abra o prompt de comando do Windows utilizando o comando windows+r digitando cmd e pressionando enter
2. Digite o comando abaixo:

```
node -v  
npm -v
```

Resultado:

```
v10.x.x  
6.x.x
```

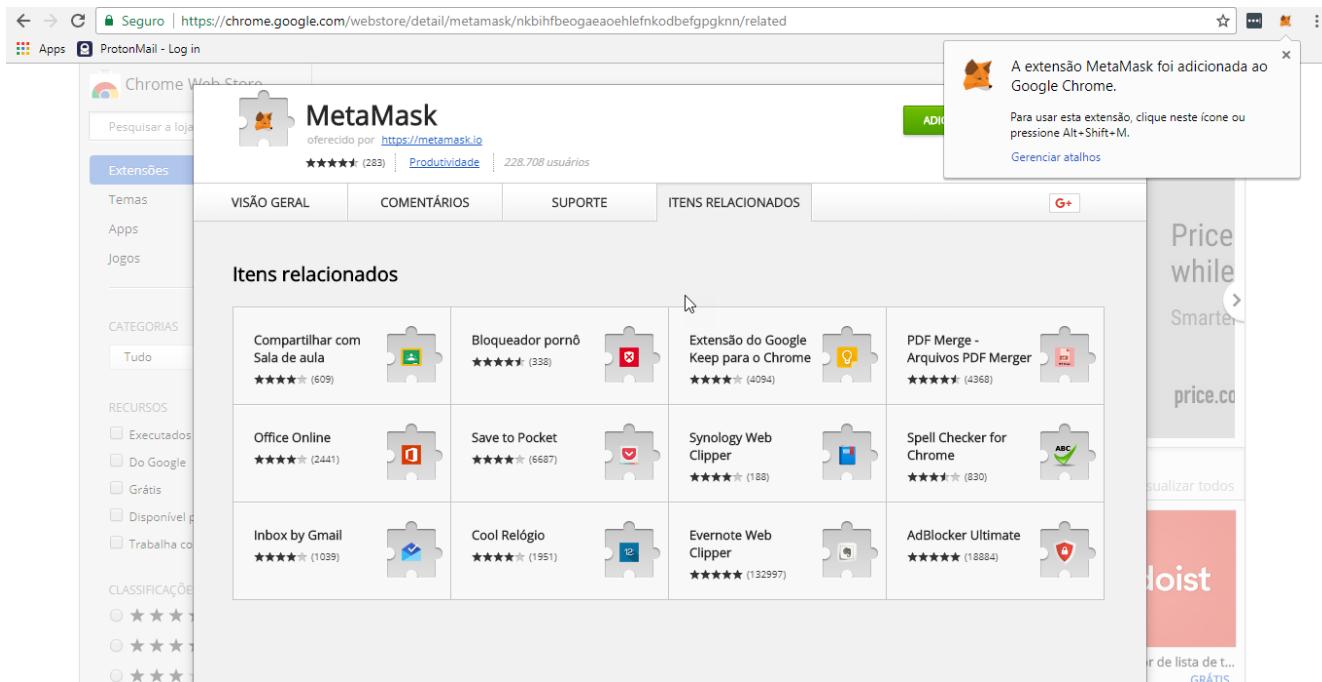
IV. MetaMask

Instalação do MetaMask

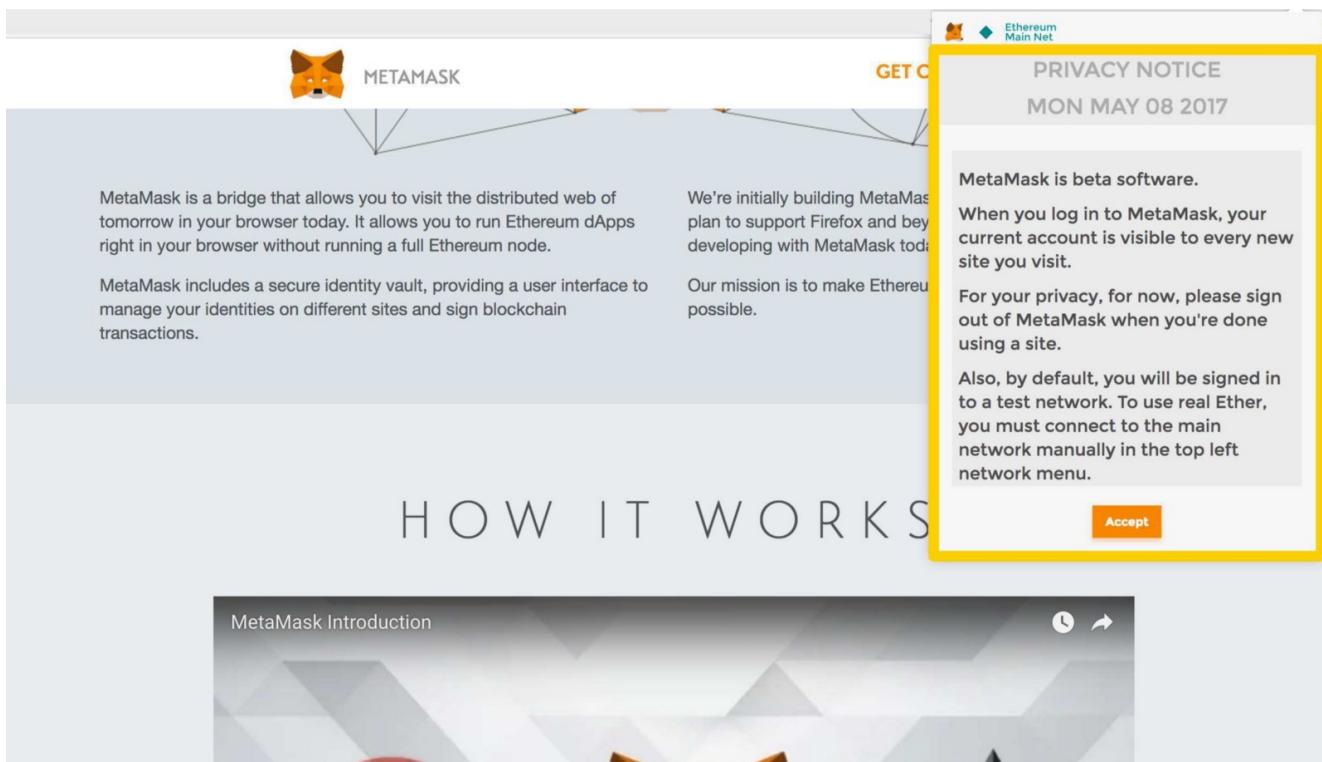
1. Acesse a página de instalação do plugin MetaMask com o navegador Google Chrome: <https://chrome.google.com/webstore/detail/metamask/nkbihfbeogaeaoehlefknkodbefgpgknn>
2. Clique em + Usar no Chrome
3. Confirme clicando em Adicionar extensão

Teste do MetaMask

1. Clique no ícone do MetaMask posicionado ao lado da barra de endereço do Google Chrome



Resultado:



V. Remix

Teste do Remix

1. Acesse a página do Remix: <http://remix.ethereum.org/>

Resultado:

The screenshot shows the Remix IDE interface. On the left, the Solidity code for `ballot.sol` is displayed. The code defines a `Voter` struct with fields `weight`, `proposal`, and `vote`, and a `Ballot` contract with methods for creating ballots, delegating votes, and voting. On the right, the browser interface shows the contract environment with an account set to `Injected Web3` and a gas limit of `3000000`. A red box highlights the `Create` button and the input field for `uint8 _numProposals`.

```
pragma solidity ^0.4.0;
contract Ballot {
    struct Voter {
        uint weight;
        address proposal;
        uint8 vote;
        address delegate;
    }
    struct Proposal {
        string label;
        uint weight;
        uint voteCount;
    }
    address chairperson;
    mapping(address => Voter) voters;
    Proposal[] proposals;
    uint8 _numProposals;
    // Create a new ballot with _numProposals different proposals.
    function createBallot(uint8 _numProposals) {
        chairperson = msg.sender;
        voters[chairperson].weight = 1;
        proposals.length = _numProposals;
    }
    // Give a voter the right to vote on this ballot.
    function giveRightToVote(address voter) {
        if (voter == chairperson || voters[voter].voted) return;
        voters[voter].weight = 1;
    }
    // Delegate your vote to the voter $(to).
    function delegate(address to) {
        Voter storage sender = voters[msg.sender]; // assigns reference
        while (voters[to].delegate != address(0) && voters[to].delegate != msg.sender)
            if (to == msg.sender) return;
        sender.voted = true;
        sender.weight += voters[to].weight;
        Voter storage delegateTo = voters[to];
        if (delegateTo.voted)
            proposals[delegateTo.vote].voteCount += sender.weight;
        else
            delegateTo.vote = msg.sender;
        delegateTo.weight += sender.weight;
    }
    // Give a single vote to proposal $(proposal).
    function vote(uint8 proposal) {
        Voter storage sender = voters[msg.sender];
        if (proposal >= proposals.length) return;
        sender.voted = true;
        sender.weight -= 1;
        proposals[proposal].voteCount += sender.weight;
    }
}
```

VI. remixd

Instalação do remixd

- Abra o prompt de comando do Windows utilizando o comando `windows+r` digitando `cmd` e pressionando `enter`
- Digite o comando abaixo:

```
npm install -g remixd
```

Teste do remixd

- Abra o prompt de comando do Windows utilizando o comando `windows+r` digitando `cmd` e pressionando `enter`
- Digite o comando abaixo:

```
remixd
```

Resultado:

```
example: --dev-path /home/devchains/chain1 --mist --geth --frontend /home/frontend --
```

```
Usage: remixd -s <shared folder>
```

Provide a two ways connection between the local computer and Remix IDE

Options:

-s, --shared-folder <path>	Folder to share with Remix IDE
-m, --mist	start mist
-g, --geth	start geth
-p, --dev-path <dev-path>	Folder used by mist/geth to start the development environment
-f, --frontend <front-end>	Folder that should be served by remixd
-p, --frontend-port <front-end-port>	Http port used by the frontend (default 8080)
-a, --auto-mine	mine pending transactions
-r, --rpc <cors-domains>	start rpc server. Values are CORS domain
-rp, --rpc-port	rpc server port (default 8545)
-h, --help	output usage information

**Caso você não obtenha o resultado esperado, acesse a pasta

%APPDATA%\npm\node_modules\remixd pelo terminal e execute o comando `npm rebuild`, após isto, teste novamente o comando `remixd`. Caso o problema ainda persista, instale a versão `LTS` do NodeJS. **

VII. OpenZeppelin

Instalação do OpenZeppelin

- Abra o prompt de comando do Windows utilizando o comando `windows+r` digitando `cmd` e pressionando `enter`
- Digite o comando abaixo:

```
cd ~
mkdir blockchain-dev
cd blockchain-dev
git clone https://github.com/OpenZeppelin/openzeppelin-solidity.git
```

Teste do OpenZeppelin

- Abra o prompt de comando do Windows utilizando o comando `windows+r` digitando `cmd` e pressionando `enter`
- Digite o comando abaixo:

```
cd ~/blockchain-dev
```

Resultado:

X. GETH

Instalação do GETH

1. Faça download do GETH em: <https://gethstore.blob.core.windows.net/builds/geth-windows-amd64-1.8.11-dea1ce05.exe>
2. Abra o arquivo `geth-windows-amd64-1.8.11-dea1ce05.exe`
3. Mova o arquivo para a pasta `blockchain-dev\dev`

Teste do GETH

1. Abra o prompt de comando do Windows utilizando o comando `windows+r` digitando `cmd` e pressionando `enter`
2. Digite o comando abaixo:

```
cd ~/blockchain-dev/geth  
geth version
```

Resultado:

```
Geth  
Version: 1.8.1-stable  
Architecture: amd64  
Protocol Versions: [63 62]  
Network Id: 1  
Go Version: go1.10  
Operating System: darwin  
GOPATH=  
GOROOT=/usr/local/opt/go/libexec
```

XI. serve

Instalação do serve

1. Abra o prompt de comando do Windows utilizando o comando `windows+r` digitando `cmd` e pressionando `enter`
2. Digite o comando abaixo:

```
npm install -g serve
```

Teste do serve

1. Abra o prompt de comando do Windows utilizando o comando `windows+r` digitando `cmd` e pressionando `enter`
2. Digite o comando abaixo:

```
serve -v
```

Resultado:

```
9.0.0
```

XII. Visual Studio Code

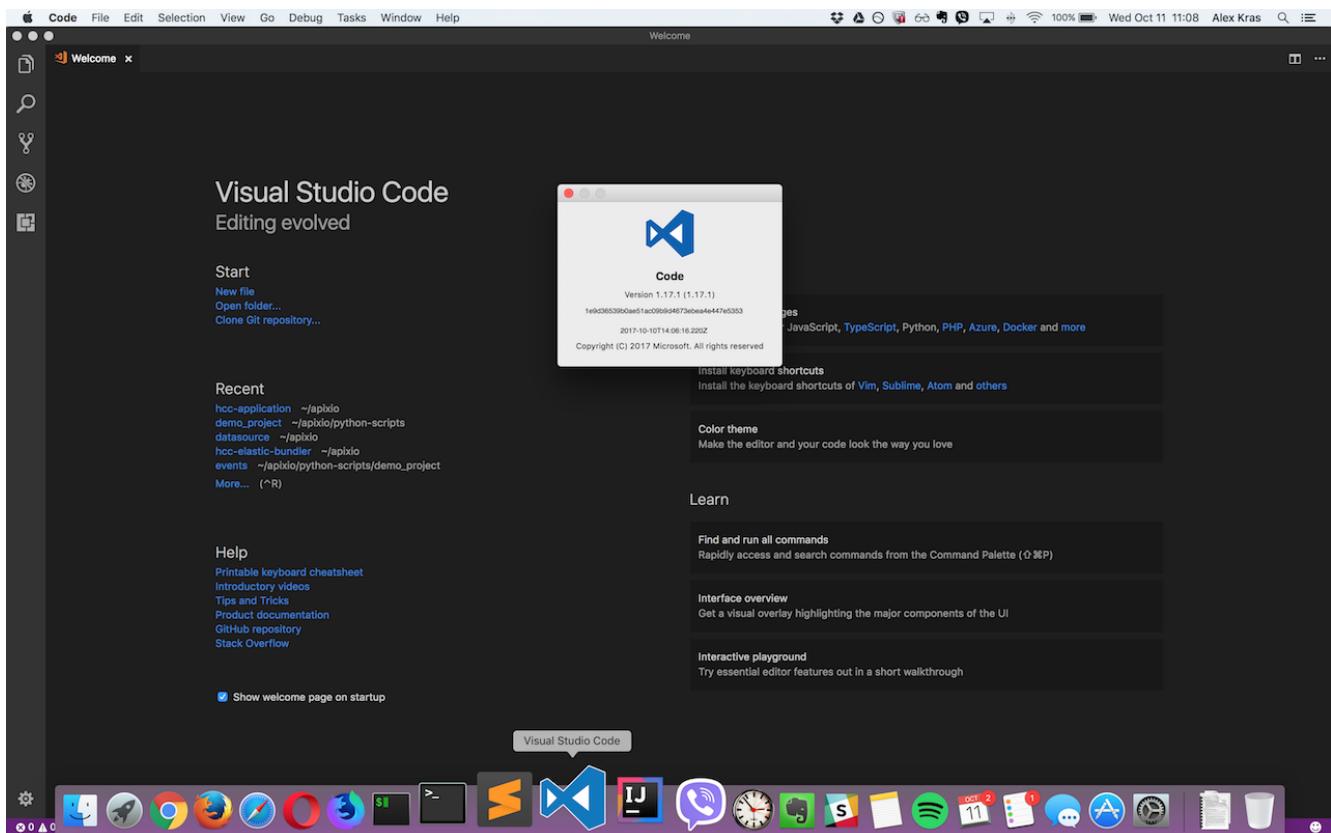
Instalação do Visual Studio Code

1. Acesse a página de instalação do Visual Studio Code:
<https://code.visualstudio.com/download>
2. Clique no botão de download para `Windows`
3. Abra o arquivo `VSCodeSetup-x64-1.24.1.exe` para descompactar
4. Não há necessidade de customização na instalação, siga o Wizard até o final.

Teste do Visual Studio Code

1. Ao final da instalação, selecione a opção para abrir o `Visual Studio Code` automaticamente.

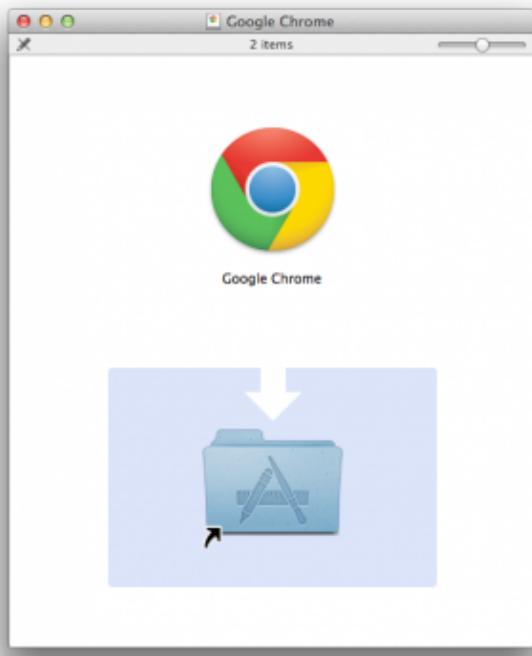
Resultado:



MacOS

I. Google Chrome

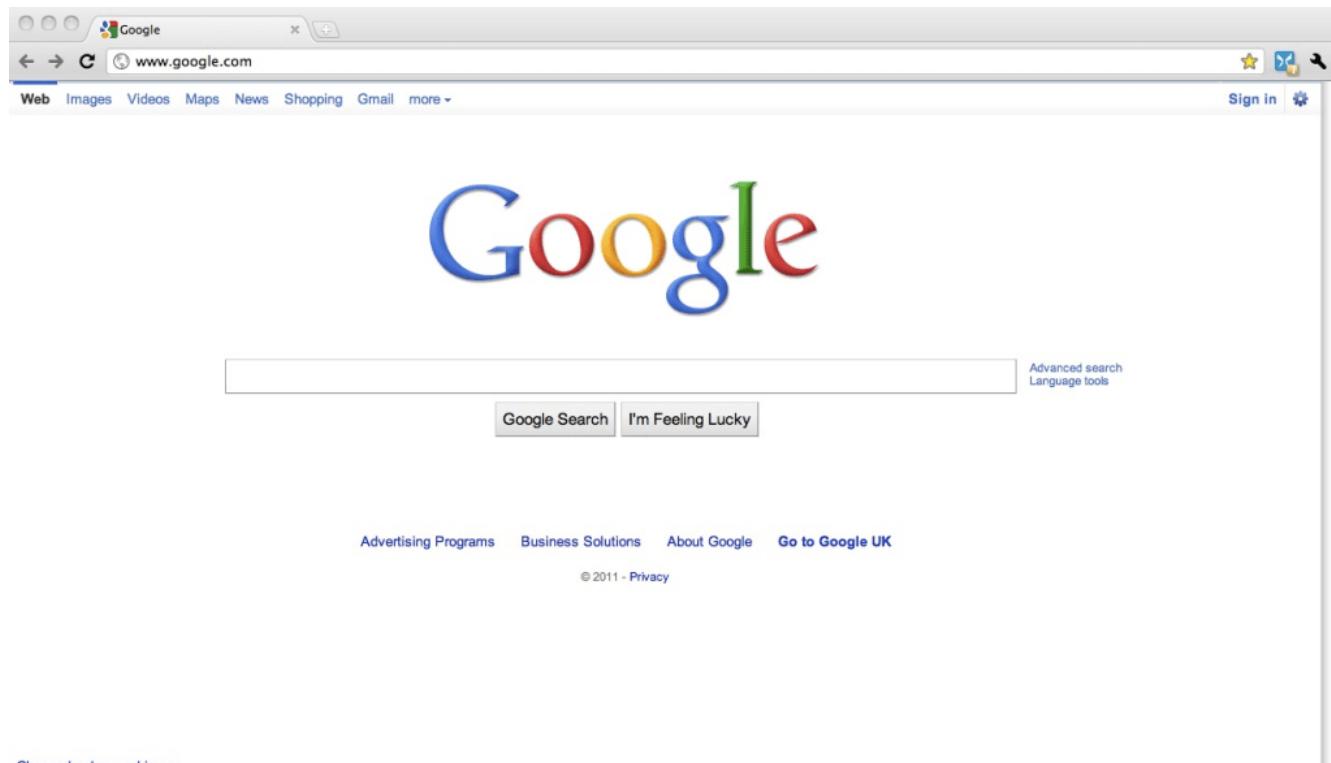
- ### Instalação do Google Chrome
1. Acesse o site do Google Chrome: <https://www.google.com/chrome>
 2. Clique em **Fazer o download do Google Chrome**
 3. Leia os termos de Serviço e clique em **Aceitar e Instalar**
 4. Abra o arquivo `googlechrome.dmg` baixado
 5. Arraste o ícone do Google Chrome para a pasta Aplicações, conforme indicado na figura abaixo



Teste do Google Chrome

1. Pressione as teclas **COMMAND** e **ESPAÇO**, simultaneamente
2. Digite **Google Chrome** e pressione **ENTER**

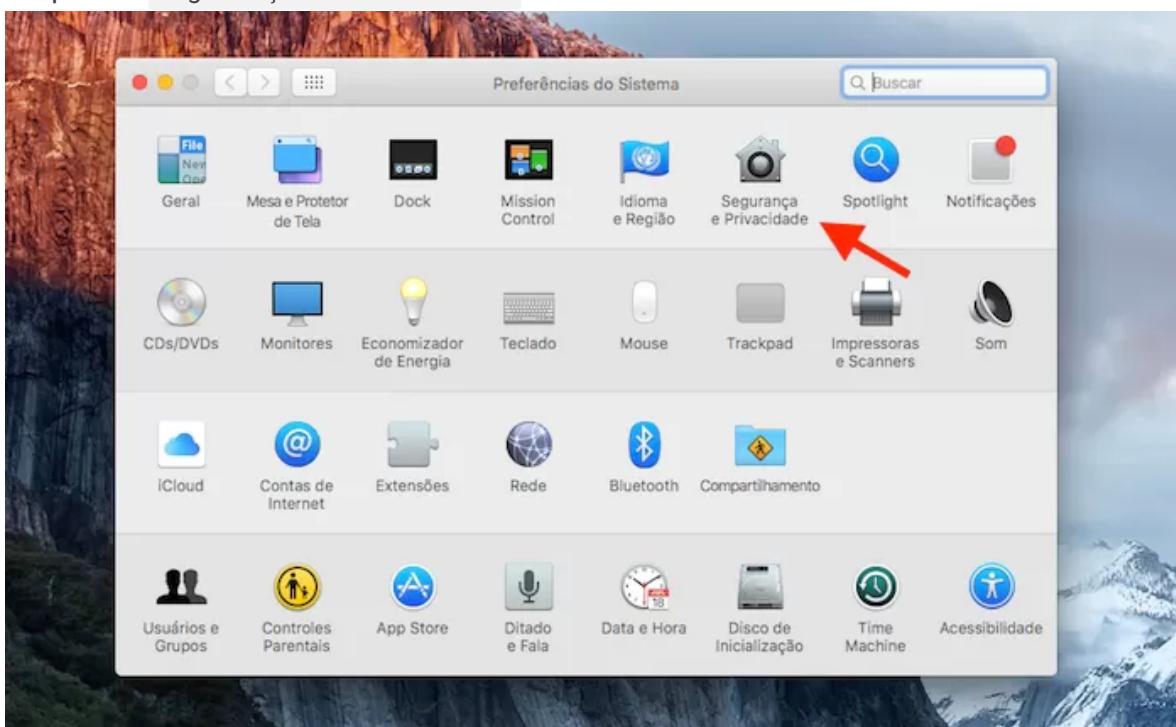
Resultado:



II. Git

Instalação do Git

1. Acesse a página de instalação do Git: <https://git-scm.com/download/mac>
2. Clique em `click here to download manually` se o download não começar em alguns instantes
3. Abra o arquivo `git-2.x.x-intel-universal-x.dmg`
4. Abra o pacote `git-2.x.x-intel-universal-x.pkg`
5. Um aviso de “desenvolvedor não identificado” será exibido. Clique em `OK`
6. Pressione as teclas `COMMAND` e `ESPAÇO`, simultaneamente
7. Digite `Preferências do Sistema` e pressione `ENTER`
8. Clique em `Segurança e Privacidade`



9. Clique no botão `Abrir Mesmo Assim`
10. Clique em `Abrir` na janela de confirmação
11. Na janela de instalação, clique no botão `Continuar` e em seguida `Instalar`
12. Digite a senha do usuário e clique em `Instalar Software`

Teste do Git

1. Pressione as teclas `COMMAND` e `ESPAÇO`, simultaneamente
2. Digite `Terminal` e pressione `ENTER`
3. No terminal de comando, digite:

```
git --version
```

Resultado:

```
git version 2.x.x
```

III. Node.js

Instalação do Node.js

1. Acesse a página de instalação do Node.js: <https://nodejs.org/en/download/>
2. Clique na aba Current - Lastes Features
3. Clique em macOS Installer
4. Abra o arquivo node-v10.x.x.pkg
5. Clique em Continuar duas vezes e Instalar
6. Digite a senha do usuário e clique em Instalar Software

Configuração do Node.js

1. Pressione as teclas COMMAND e ESPAÇO, simultaneamente
2. Digite Terminal e pressione ENTER
3. No terminal de comando, digite:

```
sudo chown -R $(whoami) ~/.npm
```

Mais detalhes em: <https://docs.npmjs.com/getting-started/fixing-npm-permissions>

Teste do Node.js

1. Pressione as teclas COMMAND e ESPAÇO, simultaneamente
2. Digite Terminal e pressione ENTER
3. No terminal de comando, digite:

```
node -v  
npm -v
```

Resultado:

```
v10.x.x  
6.x.x
```

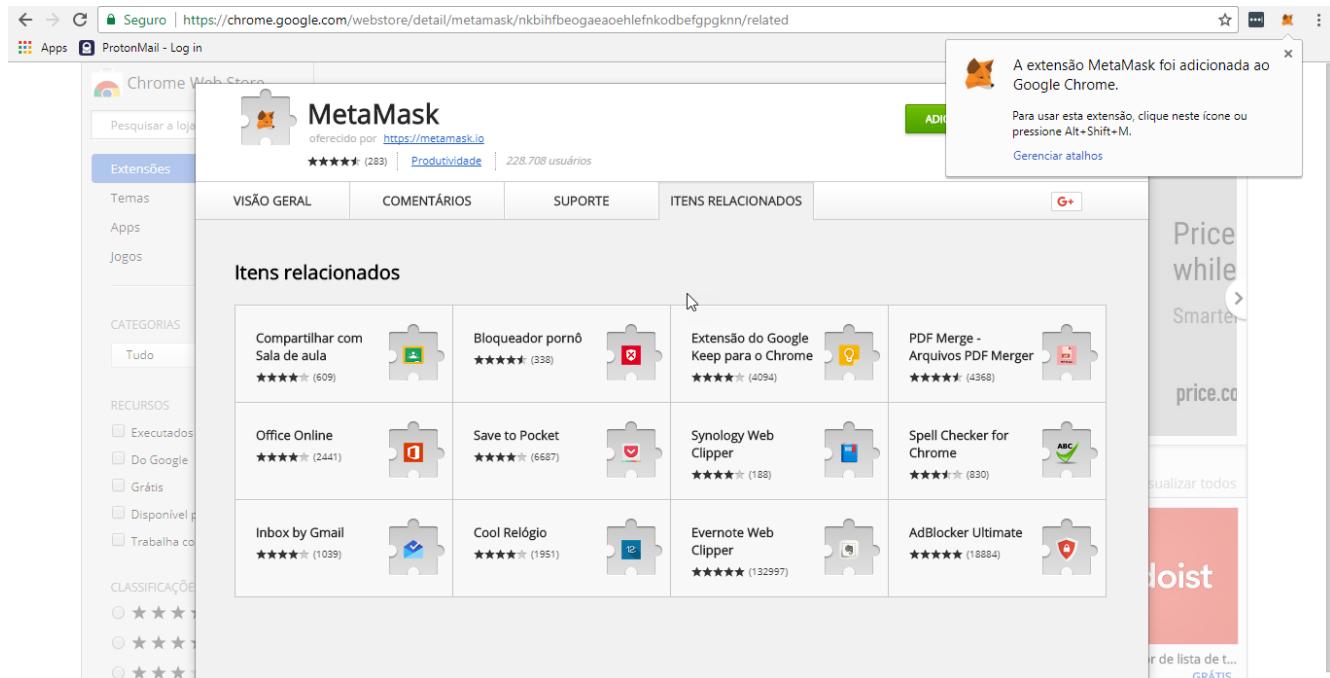
IV. MetaMask

Instalação do MetaMask

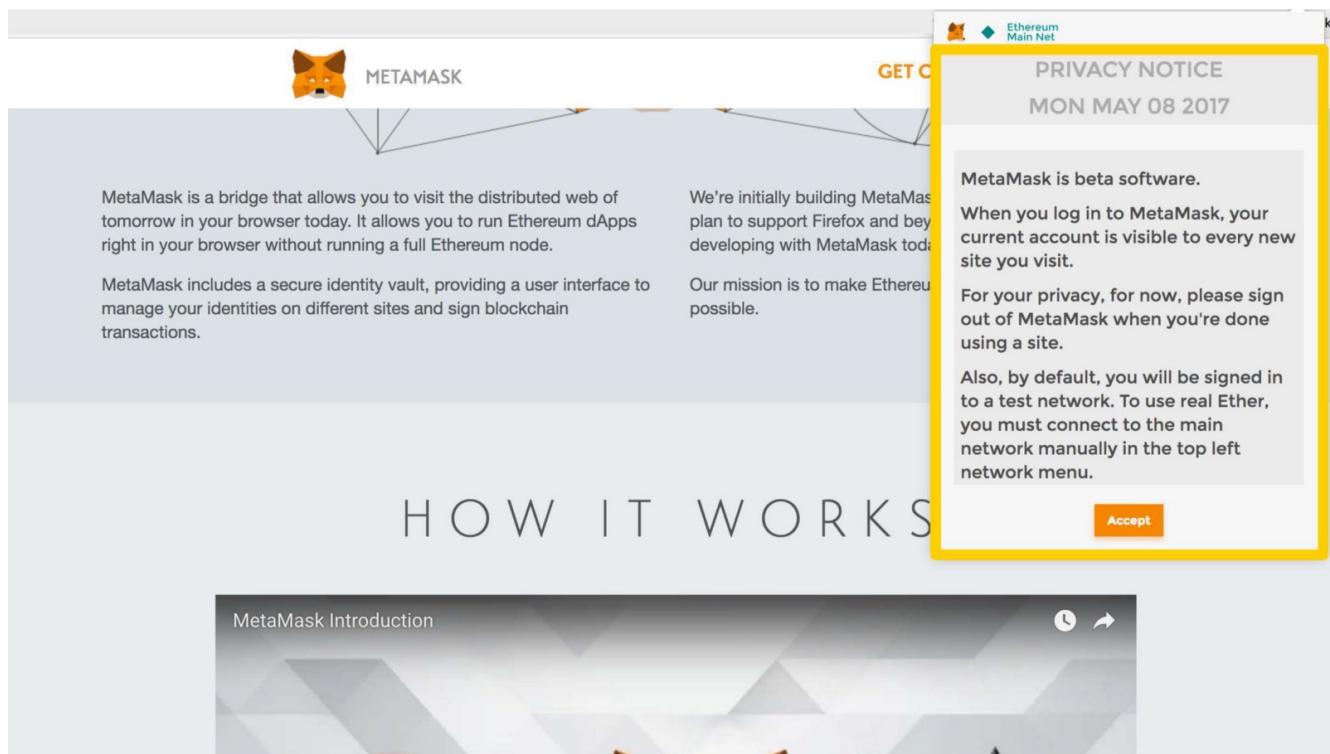
- Acesse a página de instalação do plugin MetaMask com o navegador Google Chrome:
<https://chrome.google.com/webstore/detail/metamask/nkbihfbeogaeaoehlefknkodbefgpgknn>
- Clique em **+ Usar no Chrome**
- Confirme clicando em **Adicionar extensão**

Teste do MetaMask

- Clique no ícone do MetaMask posicionado ao lado da barra de endereço do Google Chrome



Resultado:



V. Remix

Teste do Remix

1. Acesse a página do Remix: <http://remix.ethereum.org/>

Resultado:

The screenshot shows the Ethereum Remix IDE interface. On the left, the code editor displays the `ballot.sol` file with Solidity code for a ballot contract. On the right, the deployment interface shows the environment set to "Injected Web3", account selected, gas limit set to 3000000, and value set to 0. Below these settings, there is a "Create" button labeled `uint8 _numProposals`. The interface also indicates 0 pending transactions and no contract instances.

```
pragma solidity >=0.4.0;
contract Ballot {
    struct Voter {
        uint weight;
        bool voted;
        address delegate;
    }
    struct Proposal {
        uint weight;
    }
    address chairperson;
    mapping(address) voters;
    Proposal[] proposals;
    // Create a new ballot with _numProposals different proposals.
    function ballot(uint8 _numProposals) {
        for (uint i = 0; i < _numProposals; i++) {
            voters[chairperson].weight += 1;
            proposals[i].weight = 1;
        }
    }
    // Give $voter the right to vote on this ballot.
    // May only be called by $chairperson.
    function giveRightToVote(address voter) {
        if (msg.sender != chairperson || voters[voter].voted) return;
        voters[voter].weight = 0;
    }
    // Delegate your vote to the voter $to.
    function delegate(address to) {
        Voter storage sender = voters[msg.sender]; // assigns reference
        if (sender.voted) return;
        if (to == address(0) || voters[to].delegate != msg.sender)
            to = voters[to].delegate;
        if (voters[to].voted) return;
        sender.voted = true;
        Voter storage delegateTo = voters[to];
        if (delegateTo.voted)
            sender.weight -= delegateTo.voteCount;
        else
            delegateTo.voteCount += sender.weight;
        sender.weight += delegateTo.weight;
    }
    // Give a single vote to proposal $proposal.
    function vote(address proposal) {
        Voter storage sender = voters[msg.sender];
        if (sender.voted || proposal >= proposals.length) return;
        sender.voted = true;
        sender.vote = proposal;
        proposals[proposal].voteCount += sender.weight;
    }
}
```

VI. remixd

Instalação do remixd

1. Pressione as teclas `COMMAND` e `ESPAÇO`, simultaneamente
2. Digite `Terminal` e pressione `ENTER`
3. No terminal de comando, digite:

```
npm install -g remixd
```

Teste do remixd

1. Pressione as teclas `COMMAND` e `ESPAÇO`, simultaneamente
2. Digite `Terminal` e pressione `ENTER`
3. No terminal de comando, digite:

```
remixd
```

Resultado:

```
example: --dev-path /home/devchains/chain1 --mist --geth --frontend /home/frontend --
```

```
Usage: remixd -s <shared folder>
```

```
Provide a two ways connection between the local computer and Remix IDE
```

```
Options:
```

-s, --shared-folder <path>	Folder to share with Remix IDE
-m, --mist	start mist
-g, --geth	start geth
-p, --dev-path <dev-path>	Folder used by mist/geth to start the development environment
-f, --frontend <front-end>	Folder that should be served by remixd
-p, --frontend-port <front-end-port>	Http port used by the frontend (default 8080)
-a, --auto-mine	mine pending transactions
-r, --rpc <cors-domains>	start rpc server. Values are CORS domains
-rp, --rpc-port	rpc server port (default 8545)
-h, --help	output usage information

VII. OpenZeppelin

Instalação do OpenZeppelin

- Pressione as teclas `COMMAND` e `ESPAÇO`, simultaneamente
- Digite `Terminal` e pressione `ENTER`
- No terminal de comando, digite:

```
cd ~
mkdir blockchain-dev
cd blockchain-dev
git clone https://github.com/OpenZeppelin/openzeppelin-solidity.git
```

Teste do OpenZeppelin

- Pressione as teclas `COMMAND` e `ESPAÇO`, simultaneamente
- Digite `Terminal` e pressione `ENTER`
- No terminal de comando, digite:

```
ls ~/blockchain-dev
```

Resultado:

```
openzeppelin-solidity
```

X. GETH

Instalação do GETH

1. Faça download do GETH em: <https://gethstore.blob.core.windows.net/builds/geth-darwin-amd64-1.8.11-dea1ce05.tar.gz>
2. Abra o arquivo baixado `geth-darwin-amd64-1.8.11-dea1ce05.tar.gz`
3. Pressione as teclas `COMMAND` e `ESPAÇO`, simultaneamente
4. Digite `Terminal` e pressione `ENTER`
5. No terminal de comando, digite:

```
mv ~/Downloads/geth-darwin-amd64-1.8.11-dea1ce05 ~/blockchain-dev/geth
```

Teste do GETH

1. Pressione as teclas `COMMAND` e `ESPAÇO`, simultaneamente
2. Digite `Terminal` e pressione `ENTER`
3. No terminal de comando, digite:

```
cd ~/blockchain-dev/geth  
geth version
```

Resultado:

```
Geth  
Version: 1.8.1-stable  
Architecture: amd64  
Protocol Versions: [63 62]  
Network Id: 1  
Go Version: go1.10  
Operating System: darwin  
GOPATH=  
GOROOT=/usr/local/opt/go/libexec
```

Veja mais detalhes em <https://github.com/ethereum/go-ethereum/wiki/Installation-Instructions-for-Mac>

XI. serve

Instalação do serve

1. Pressione as teclas **COMMAND** e **ESPAÇO**, simultaneamente
2. Digite **Terminal** e pressione **ENTER**
3. No terminal de comando, digite:

```
npm install -g serve
```

Teste do serve

1. Pressione as teclas **COMMAND** e **ESPAÇO**, simultaneamente
2. Digite **Terminal** e pressione **ENTER**
3. No terminal de comando, digite:

```
serve -v
```

Resultado:

```
9.0.0
```

XII. Visual Studio Code

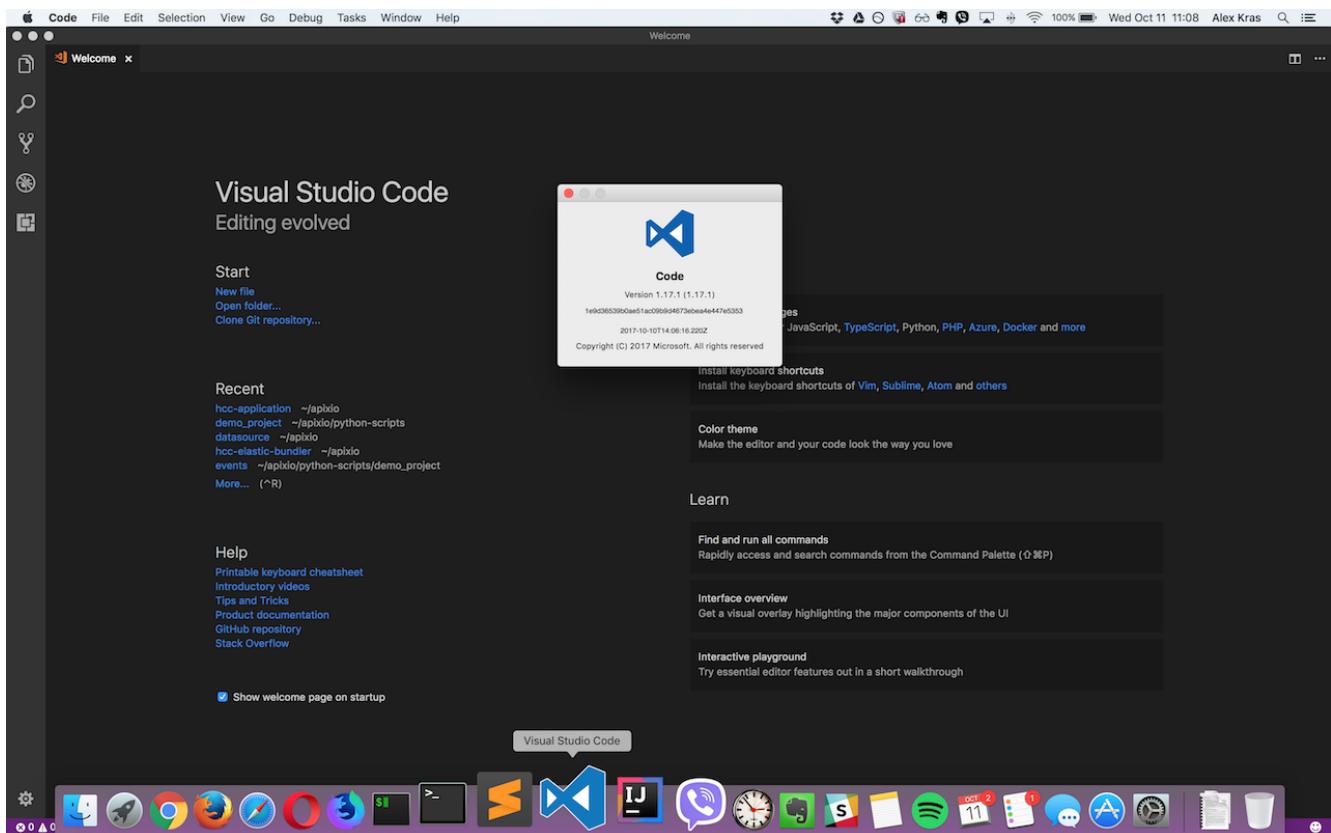
Instalação do Visual Studio Code

1. Acesse a página de instalação do Visual Studio Code:
<https://code.visualstudio.com/download>
2. Clique no botão de download para **Mac**
3. Abra o arquivo **VSCode-darwin-stable.zip** para descompactar
4. Arrate o arquivo descompactado **Visual Studio Code** para a pasta **Aplicativos**

Teste do Visual Studio Code

1. Pressione as teclas **COMMAND** e **ESPAÇO**, simultaneamente
2. Digite **Visual Studio Code** e pressione **ENTER**

Resultado:



Linux

As instruções de instalação descritas abaixo foram elaboradas com base no Ubuntu 18.04 LTS. Instruções diferentes ou adicionais podem ser necessárias dependendo da distribuição do Linux utilizada.

I. Google Chrome

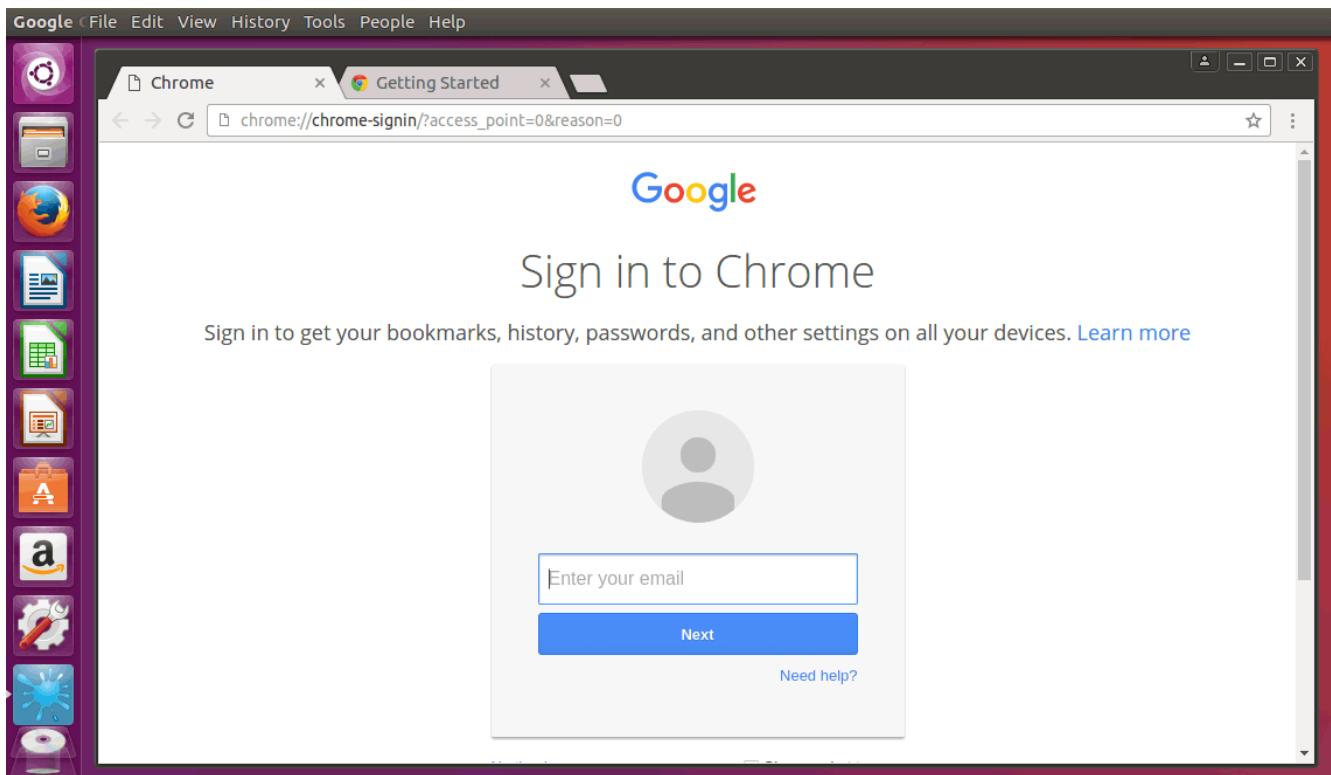
Instalação do Google Chrome

1. Acesse o site do Google Chrome: <https://www.google.com/chrome>
2. Clique em **Download Chrome**
3. Leia os termos de Serviço e clique em **Accept and Install**
4. Abra o arquivo `google-chrome-stable_current_amd64.deb` baixado
5. Clique no botão **Instalar**

Teste do Google Chrome

1. Clique no ícone **Mostrar aplicativos**
2. Pesquise por **Google Chrome** e abra o aplicativo

Resultado:



II. Git

Instalação do Git

1. Abra o terminal de comandos e digite:

```
sudo apt-get update  
sudo apt-get install -y git-core
```

Teste do Git

1. Abra o terminal de comandos e digite:

```
git --version
```

Resultado:

```
git version 2.x.x
```

III. Node.js

Instalação do Node.js

1. Abra o terminal de comandos e digite:

```
sudo apt install -y curl
curl -sL https://deb.nodesource.com/setup_10.x | sudo -E bash -
sudo apt-get install -y nodejs
sudo apt-get install -y build-essential
```

Configuração do Node.js

1. Abra o terminal de comandos e digite:

```
mkdir ~/.npm-global
npm config set prefix '~/.npm-global'
export PATH=~/.npm-global/bin:$PATH
```

Mais detalhes em: <https://nodejs.org/en/download/package-manager/>

Teste do Node.js

1. Abra o terminal de comandos e digite:

```
node -v
npm -v
```

Resultado:

```
v10.x.x
6.x.x
```

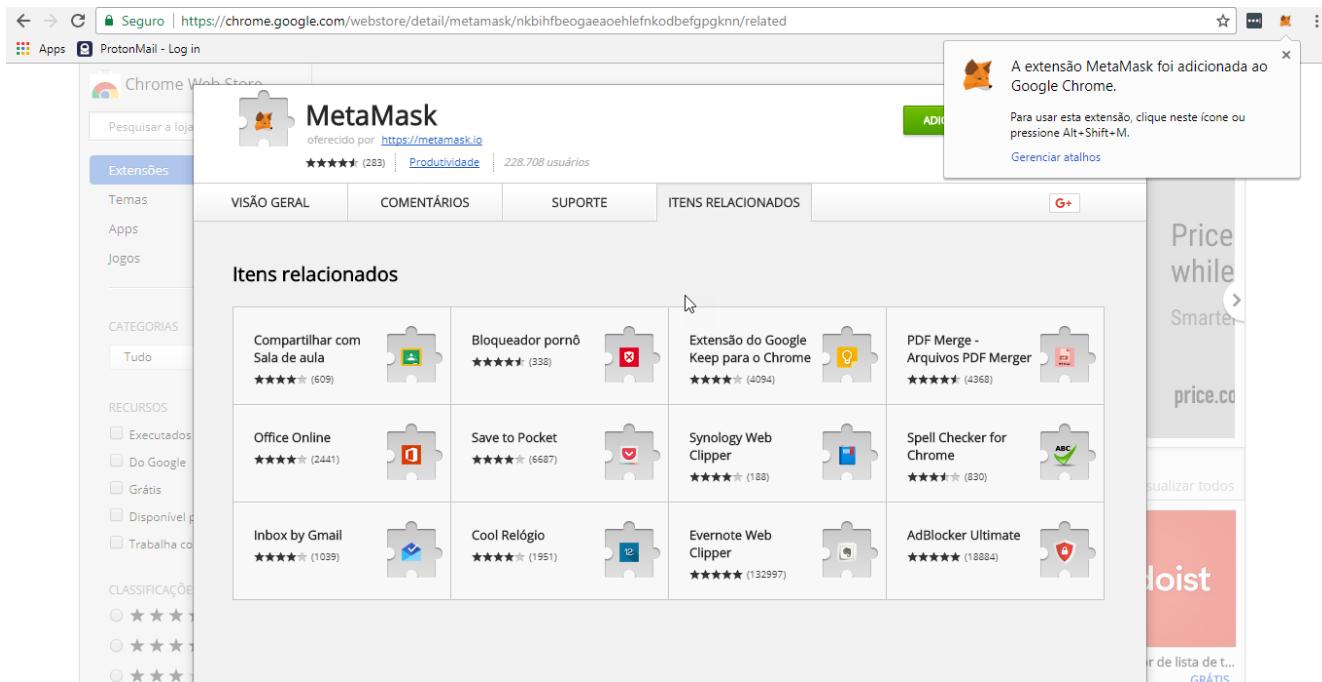
IV. MetaMask

Instalação do MetaMask

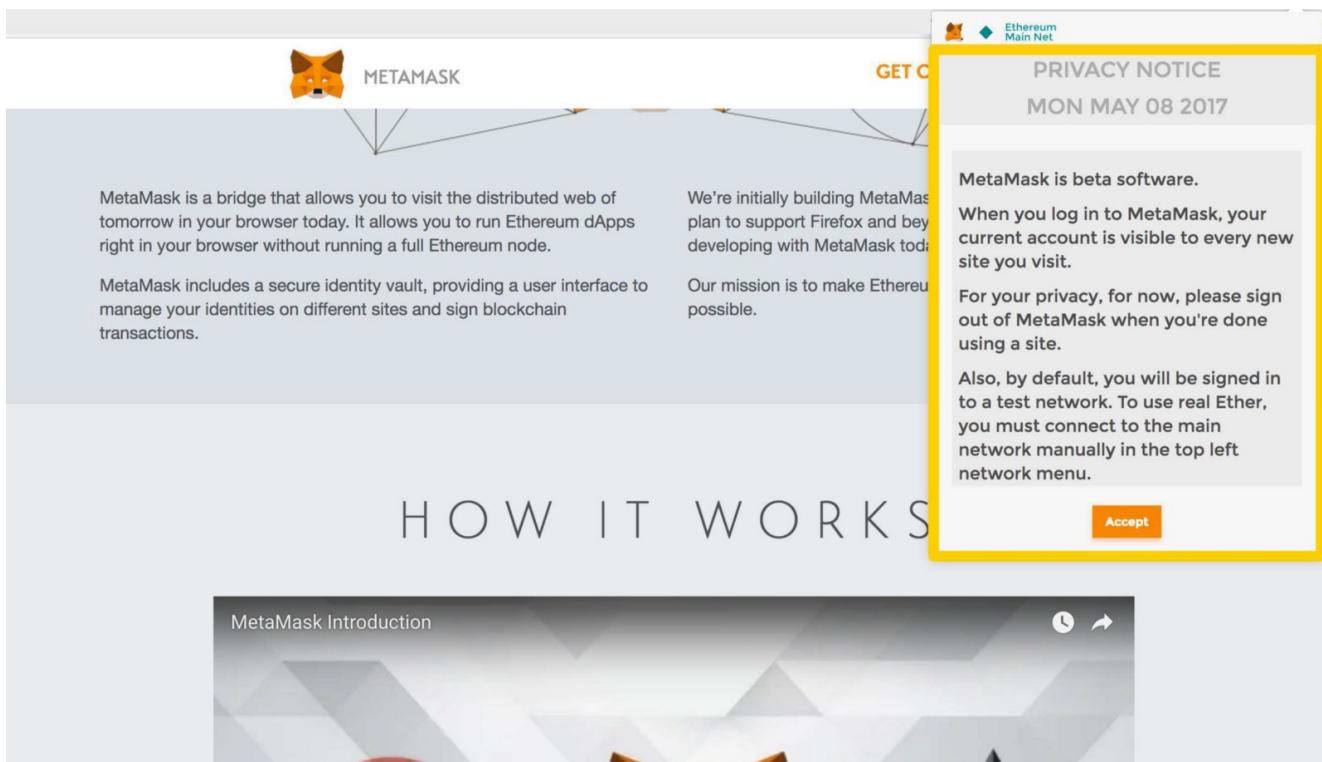
1. Acesse a página de instalação do plugin MetaMask com o navegador Google Chrome:
<https://chrome.google.com/webstore/detail/metamask/nkbihfbeogaeaoehlefknkodbefgpgknn>
2. Clique em **+ Usar no Chrome**
3. Confirme clicando em **Adicionar extensão**

Teste do MetaMask

1. Clique no ícone do MetaMask posicionado ao lado da barra de endereço do Google Chrome



Resultado:



V. Remix

Teste do Remix

1. Acesse a página do Remix: <http://remix.ethereum.org/>

Resultado:

The screenshot shows the Remix IDE interface. On the left, the code editor displays the `ballot.sol` file. The code defines a `Voter` struct with fields `weight`, `proposal`, and `vote`, and a `Ballot` contract with a `Proposals[] proposals` array. It includes functions for creating a ballot, giving delegation rights, and voting. On the right, the deployment interface shows the `browser/ballot.sol:Ballot` contract with a `Create` button and a `Value` input field set to 0. Below the interface, a message says "No Contract Instances".

```
< browser/ballot.sol >
browser
ballot.sol
pragma solidity ^0.4.0;
contract Ballot {
    struct Voter {
        uint weight;
        address proposal;
        uint8 vote;
        address delegate;
    }
    struct Proposal {
        uint id;
        string label;
        uint weight;
        uint tally;
    }
    mapping(address => Voter) voters;
    Proposal[] proposals;
    // Create a new ballot with 5 different proposals.
    function createBallot(uint8 _numProposals) {
        chairperson = msg.sender;
        voter[chairperson].weight = 1;
        proposals.length = _numProposals;
    }
    // Give a voter the right to vote on this ballot.
    function giveRightToVote(address voter) {
        if (voter == chairperson || voters[voter].voted) return;
        voter[voter].weight = 1;
    }
    // Delegate your vote to the voter $(to).
    function delegate(address to) {
        Voter storage sender = voters[msg.sender]; // assigns reference
        while (voter[to].delegate != address(0) && voters[to].delegate != msg.sender)
            to = voters[to].delegate;
        if (to == msg.sender) return;
        sender.voted = true;
        voter[to].delegated = true;
        Voter storage delegateTo = voters[to];
        if (delegateTo.voted) {
            proposals[delegateTo.vote].tally += sender.weight;
        } else {
            proposals[delegateTo.vote].tally = sender.weight;
        }
        delegateTo.weight += sender.weight;
    }
    // Give a single vote to proposal $(proposal).
    function vote(uint8 proposal) {
        Voter storage sender = voters[msg.sender];
        if (proposal >= proposals.length) return;
        sender.voted = true;
        voter[msg.sender].proposal = proposal;
        proposals[proposal].voteCount += sender.weight;
    }
}
```

VI. remixd

Instalação do remixd

- Abra o terminal de comando e digite:

```
npm install -g remixd
```

Teste do remixd

- Abra o terminal de comando e digite:

```
remixd
```

Resultado:

```
example: --dev-path /home/devchains/chain1 --mist --geth --frontend /home/frontend --
```

Usage: `remixd -s <shared folder>`

Provide a two ways connection between the local computer and Remix IDE

Options:

`-s, --shared-folder <path>`

Folder to share with Remix IDE

```
-m, --mist                      start mist
-g, --geth                       start geth
-p, --dev-path <dev-path>        Folder used by mist/geth to start the deve]
-f, --frontend <front-end>       Folder that should be served by remixd
-p, --frontend-port <front-end-port> Http port used by the frontend (default 8080)
-a, --auto-mine                  mine pending transactions
-r, --rpc <cors-domains>        start rpc server. Values are CORS domain
-rp, --rpc-port                  rpc server port (default 8545)
-h, --help                        output usage information
```

VII. OpenZeppelin

Instalação do OpenZeppelin

1. Abra o terminal de comando e digite:

```
cd ~
mkdir blockchain-dev
cd blockchain-dev
git clone https://github.com/OpenZeppelin/openzeppelin-solidity.git
```

Teste do OpenZeppelin

1. Abra o terminal de comando e digite:

```
ls ~/blockchain-dev
```

Resultado:

```
openzeppelin-solidity
```

X. GETH

Instalação do GETH

1. Faça download do GETH em: <https://gethstore.blob.core.windows.net/builds/geth-linux-amd64-1.8.11-dea1ce05.tar.gz>
2. Abra o arquivo baixado `geth-linux-amd64-1.8.11-dea1ce05.tar.gz` e descompacte o conteúdo em `~/blockchain-dev/`
3. Renomeie o diretório `geth-linux-amd64-1.8.11-dea1ce05` para `geth`

Teste do GETH

1. Abra o terminal de comando e digite:

```
cd ~/blockchain-dev/geth  
.geth version
```

Resultado:

```
Geth  
Version: 1.8.1-stable  
Architecture: amd64  
Protocol Versions: [63 62]  
Network Id: 1  
Go Version: go1.10  
Operating System: darwin  
GOPATH=  
GOROOT=/usr/local/opt/go/libexec
```

XI. serve

Instalação do serve

1. Abra o terminal de comando e digite:

```
npm install -g serve
```

Teste do serve

1. Abra o terminal de comando e digite:

```
serve -v
```

Resultado:

```
9.0.0
```

XII. Visual Studio Code

Instalação do Visual Studio Code

1. Acesse a página de instalação do Visual Studio Code:
<https://code.visualstudio.com/download>
2. Clique no botão de download `.deb`, para Ubuntu
3. Abra o arquivo `code_1.x.x-x_amd64.deb`
4. Clique no botão `Instalar`

Teste do Visual Studio Code

1. Clique no ícone `Mostrar aplicativos`
2. Pesquise por `Visual Studio Code` e abra o aplicativo

Resultado: