ToyBlock

Webpage explaining the concept of blockchain in a pedagogic way.

User Guide

Juin 2022

Antoine Van Aelst

Contents

1 Introduction

1.1 What is ToyBlock

ToyBlock is a webpage popularizing how the blockchain technology works in a pedagogic way. The user follows the story of an animal village on their way to create a new currency and learns how to build a **blockchain technology**, step by step by resolving all the problems encountered by the animals of the village.

1.2 Deployed version

You can check a deployed version of this webpage at this address:
https://ushien.github.io/ToyBlock/

1.3 Author

This project is developed by Antoine Van Aelst as part of the Unamur *INFOB318 Projet Individuel* class.

Author: Antoine Van Aelst
Project manager: Jérôme Fink
Teacher in charge: Vincent Englebert

2 About this guide

2.1 Purpose

Since writing a guide about a straightforward story-structured popularizing website isn't relevant, the purpose of this guide is to help the user to create their own website starting from the structure of ToyBlock. An additional guide about the different machines of the project is also provided.

2.2 Requirements

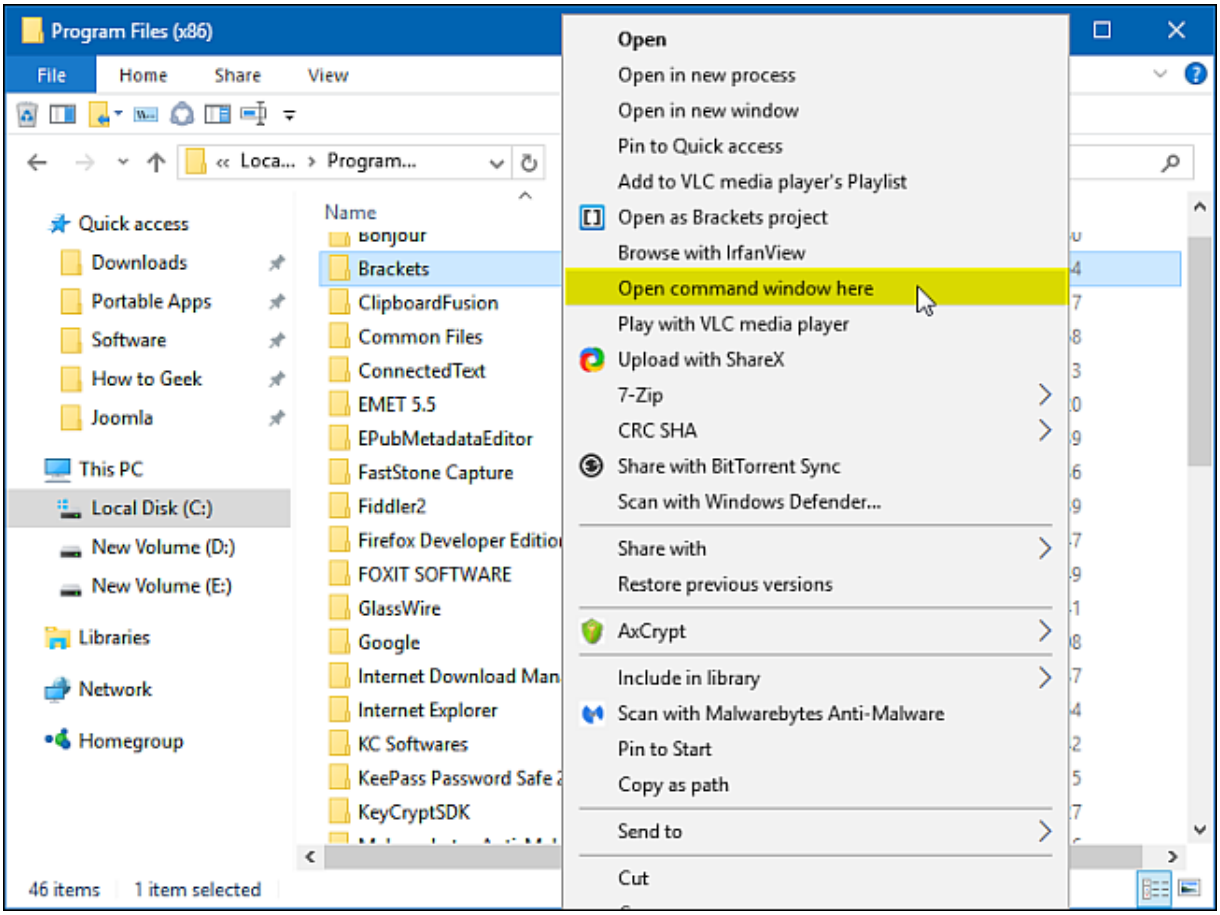
No previous experience with development is required to read this guide. This tutorial will offer the reader the first basic tools needed to understand how to improve an application from a given structure.
The only requirements for this guide are a **functional computer**, running on a recent version of Windows, macOS or Linux, and an **internet connection**.

3 Getting Started

3.1 Open a command prompt

Multiple steps of this guide require to interact with a command prompt. Here is the way to open it.

First, navigate to the *ToybBlock/code/toyblock* directory. Now open the command prompt. On Windows, you just need do press **shift+right click** and then click on *Open command window here*.



This will open the command prompt, in which you can type all sorts of useful commands to manage your program.

3.2 Install the required software

To follow this user guide you'll need to have multiple applications and libraries installed on your computer. Here are the recommended ones but many other

options are possible depending on your needs.

- **Archive file extractor**: WinRAR
- **Code editor**: VSCode
- **JavaScript runtime environment**: Node.js

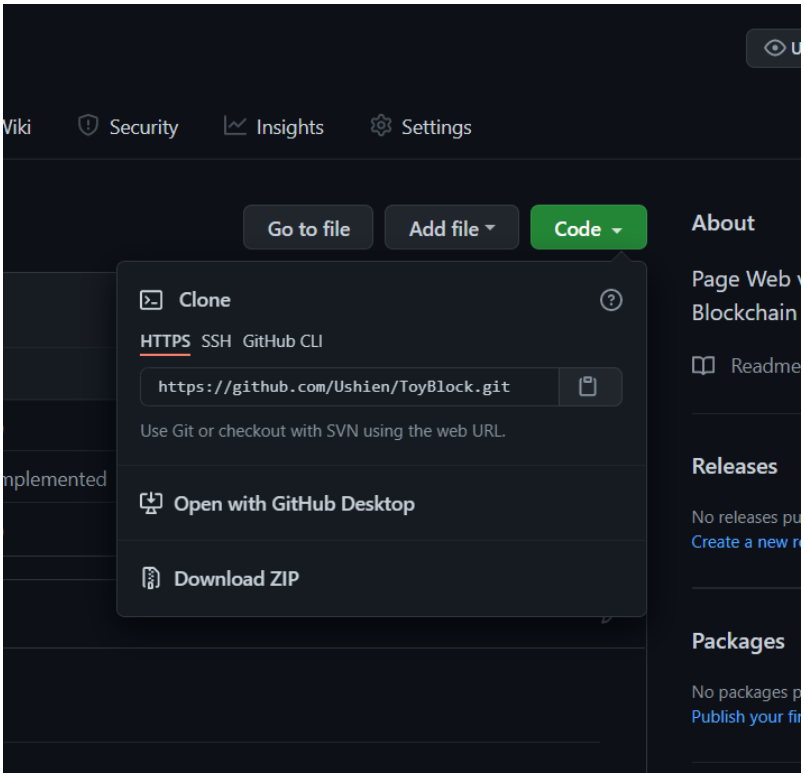
Also install the necessary library by opening a command prompt and typing this command:

```
npm install react-scripts --save
```

Be careful to download and install Node.js before typing this command.

3.3 Get the code

To get the code, you need to download it on your computer. Go on the dedicated GitHub webpage, and then click on the green button. This will display a few options.



Click on the *Download ZIP* option and wait for the end of the download.

Now that the file is downloaded we need to extract it.

Open the file with **WinRAR** or any similar program and extract it anywhere you want on your computer.

4 Guide

4.1 How to test the code

Testing the code of the program is extremely simple. First you need to open a command prompt. Learn how to open one here. Now just type the following command:

```
npm start
```

If the syntax of your code is correct, it will automatically open a tab in your navigator containing a test version of the program. Note that only **you** have access to this page and nobody else can consult it until you deploy it.

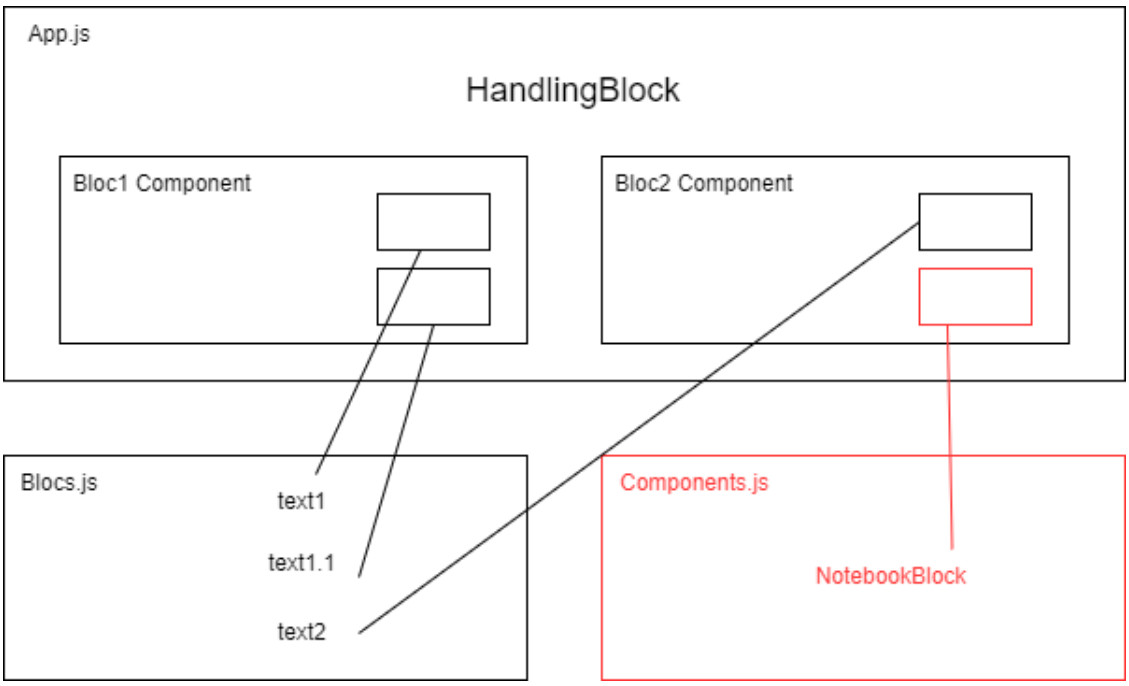
You can now edit the code in any way you want. You just need to save your file with **ctrl+s** and it will automatically apply the changes and reload the page.
Please remember that if you shut your command prompt down, you'll need to type the command again the next time you want to test your code.

4.2 How does the code work ?

To modify the website you'll only need to work with 4 files:

1. App.js: Contains the structure and the main components of the project.
2. Blocs.js: Contains the whole text of the website distributed into different blocks.
3. Components.js: Contains the components coding the different interactive machines within the story.
4. Classes.js: Contains the JavaScript classes with which the different machines interact

Here is a summary of the structure of the program:



The story is structured with a number called **distance**. This number represents the progress the user made into the story. To make this possible, we have to look into the **HandlingBlock** component of the code.

```
if (this.state.distance === 1){···
}
if (this.state.distance ===2){
  console.log("Entered Intro Block 2")
  return (
  <div>
    <BlocIntro2 onDistanceChange={this.incrementDistance} distance={this.state.distance}/>
  </div>
  )
}
if (this.state.distance ===3){···
}
```

This structure represents what the page has to display for every step of the story. Each step refers to an associate **Block** component telling it what should exactly be displayed.

The component blocks can be found by scrolling down the App.js file.

4.3 How to extend the story

You can easily add a new progress step to the story by following this method.

First, go to the **Blocks.js** file and create a new function following this template:

```
function yourname(){
        return(<div>
                Your text
        </div>
        )
}
```

Don't forget to replace *yourname* with any name you want, and replace *Your text* with the text you want to add to the story.
Secondly, add the name you chose **at the end of the document**, inside of the brackets.

```
export {introtext1, introtext2, yourname}
```

Now go to **App.js**, and create your own Block component, following this template.

```
class yourblockname extends Component {

  render(){
    return(
      <div>
        {yourname()}
      </div>
    )
  }

}
```

Also, don't forget to add the name you chose at the beginning of the App.js file.

```
import { introtext1, introtext2, yourname } from './Blocs.js';
```

The last thing left is to create a condition inside of the **HandlingBlock** conponent, a few lines further.

```
if (this.state.distance >= 14){
  wholetext.push(<div>
      <yourblockname/>
    </div>)
}
```

The new text you added should now display after the rest of the story.

5 Machine components

5.1 How to extend the story

You can also add one of the pre-implemented machines of the program into the website, including **Transaction**, **Notebook**, and **Village** machines.
For this purpose, make sure to understand the structure of the machines components. The current version of the software defines 3 types of objects to represent the story:

- A **Transaction** represents a transaction of money from one villager to another
- A **Notebook** is held by one villager and represents a set of Transactions among the villagers.
- A **Village** represents a set of Notebooks held by multiple villagers.

Then, go to the **App.js** file and add a new component following these templates. To add a village machine:

```
<div class="marged centeredtext">
  <VillageBlock
    basemoney={15}
    animals={animals}
    neighbors={neighbors}
    fillEmptyTransaction={true}

    limit={8}
    resettable={true}
    moneyName={this.props.moneyname}
  />
</div>
```

Here are the parameters you can modify:

1. **basemoney** : Defines the base amount of money of every villager.
2. **limit** : Defines the maximum amount of transactions of every notebook.
3. **resettable** : Defines if the machine is resettable or not.

Make sure to leave the other parameters unchanged.

To add a notebook machine:

```
<div class="marged centeredtext">
  <NotebookBlock
    notebook={notebook}
    limit={6}
    resettable={true}
    inVillage={false}
    moneyName={this.props.moneyname}
  />
```

Here are the parameters you can modify:

1. **limit** : Defines the maximum amount of transactions of every notebook.
2. **resettable** : Defines if the machine is resettable or not.

Additionally, define a village object before the return instruction, following this syntax.

```
let notebook = new Notebook("Toucan", 15, animals, true)
```

Here are the parameters you can modify:

1. The first parameter defines the name of the villager owning the notebook.
2. The second parameter defines the base amount of money of every villager.

Make sure to leave the other parameters unchanged too.

You can now display your own interactive scenarios inside of the story.

6 Troubleshooting

6.1 The program doesn't start and displays error messages

- Try to use **ctrl+z** command to get back to a previous state of your code and understand which command created the bug.
- Try to understand what the error message tells and research it on platforms like Stack Overflow.
- Use tools like Github to keep track of your code, and easily cancel any problematic modification.

6.2 You clicked the 'Download ZIP' button but nothing appeared in the Download folder

You may have a different destination folder than the default one. Check the Downloads section of your navigator to find the exact place of the file.
If your navigator features are limited by someone or by an organization, you also might be unable to download files from certain sources.

6.3 The instructions typed in the command prompt don't seem to work.

First make sure that you correctly spelled the instruction.
Secondly, check that you opened your command prompt at the good place. The place where you open your prompt change the behaviour of the instructions.
Finally, make sure you press the **Enter** button after any instruction you want to run.

7 FAQ

7.1 Does the created website work on both computer and mobile navigators ?

Yes it does. However some interactive features might slightly differ depending on the platform so make sure to read the associate documentation before adding anything.

7.2 I would like to connect different pages to my current page, with different URL. Is it possible ?

What you need here is routing technology. This kind of mechanic is not usable with the current structure of the program. If you need to implement it please check the react-router library.

7.3 I want to add colors and illustrations to the program, what can I do ?

The project contains a app.css file that will not be covered in this guide. Please consider checking the CSS documentation to learn how to visually modify the website. Note that every additional image file needed should be added to the "visual" folder.

8 Contact

Any question or remark ? Send an email to antoinevaelst@gmail.com.