Data Visualisation of Ethereum Currency

Kittitat Bamrung

N00201327

Date submitted: 08/03/22

Creative Coding Part 2

CA 2 – Bar Charts Project

Year 2 2021-22

DL836 BSc (Hons) in Creative Computing

Contents

Methodology:

These are all the method utilised in the Continuous Assignment (CA):

If and Else Conditional Statements

The If and Else functions are both usable in the p5 library and classic JavaScript (JS) and could be called to check if a statement is true or false i.e. a Boolean. The statement also needs to be placed between the parenthesis, in order for the function to run without failed. These conditional statements will be seen throughout my code as I have to used them to show and hide my chart labels and values (fig 1).

```
// display bar values on top of each bar
if (this.showBarValue) {
    textSize(this.valueSize);
    text(this.data[i].value, this.scaleDataHorizontal(this.data[i].value) + this.valueSize*2, -this.barWidth * i - (this.barSpacing*i) - (this.barWidth / 2));
}
// display bar labels on bottom of each bar
if (this.showLabel) {
    textSize(this.labelSize);
    text(this.data[i].label, -this.sideMargin, -this.barWidth * i - (this.barSpacing*i) - (this.barWidth / 2));

}
```

Figure 1 If and Else statement to show/hide bar values and labels

They are also implemented to do the bar animation (fig 3) which increase the data's value from 0 to a value of the bar.

```
if (this.data[i].openPriceAnimate < this.data[i].openPrice + (this.data[i].openPrice - this.data[i].closePrice)) {
    this.data[i].openPriceAnimate += 30;
}
```

Figure 2 An if statement to animate the bars

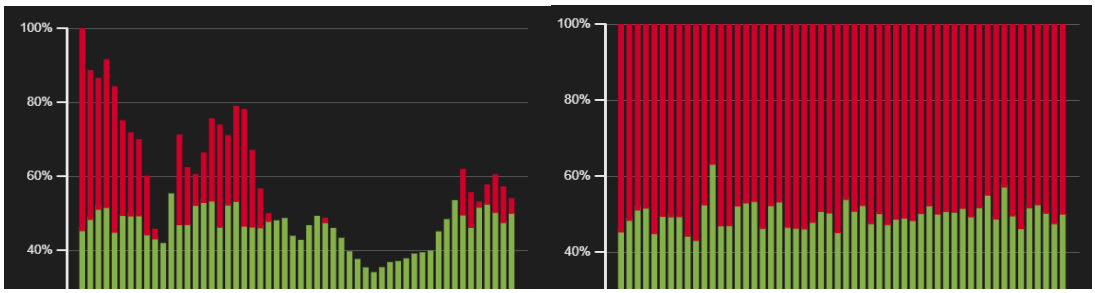

Figure 3 The translating animation(left) and resulting animation (Right)

Min() and Max()

The min() and max() functionality are accessible when importing the p5 library into the workspace. The method simply behaves as an opt that ran through the given array and pick out if either the largest or smallest value in the array.

```
this.maxValue = max(listValues);
```

Figure 4 maxValue variable

Taken this statement in the candle stick chart for instance, the variable is storing the max value (fig 4) in the listValues array.

```
4891.70459          candleStickChart.js:52
```

The output result (fig 5) can be seen above which logged the largest number in the set.

For loop

The implementations of the For Loop could be seen re-occurring throughout the libraries of charts among different classes in my CA. For Loop is a P5 function which iterates through statements by the amount of time declared in the parameter. This method is used in order to extract various data to the extent of encapsulating it to a number of functions such as drawBar() method (fig 6) in the horizontal bar chart class. The class draws each bar by a specific amount received from those extracted data by giving the loop the length of the data, in this case would be the length of the array, which in turn would allow me to also add spacing between bars and display its values on top of the given bars.

```
for (let i = 0; i < this.data.length; i++) {
    textAlign(RIGHT, CENTER);

    noStroke();
    fill(tickColor);
    // display bar values on top of each bar
    if (this.showBarValue) {
        textSize(this.valueSize);
        text(this.data[i].value, this.scaleDataHorizontal(this.data[i].value) + this.valueSize*2, -this.barWidth * i - (this.barSpacing*i) - (this.barWidth / 2));
    }
    // display bar labels on bottom of each bar
    if (this.showLabel) {
        textSize(this.labelSize);
        text(this.data[i].label, -this.sideMargin, -this.barWidth * i - (this.barSpacing*i) - (this.barWidth / 2));

    }
    // modulus is used to looped through limited colour set
    noStroke();
    fill(colors[i%4]);
    // draw bars
    rect(0, -this.barWidth * i - (this.barSpacing*i), this.scaleDataHorizontal(this.data[i].value), -this.barWidth);
}
```

Figure 6 The for loop iterating through statements

The resulting chart (fig 7) is presented below.

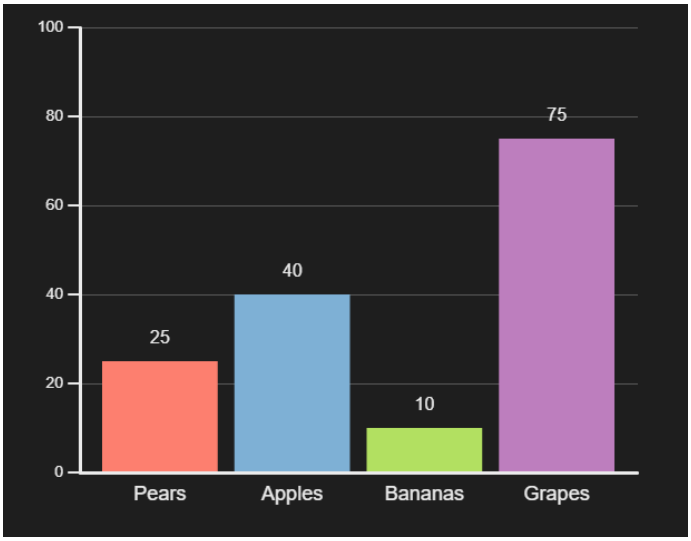

Figure 7 Resulting chart

ArrayName.map()

This function is the classic method available for usage in the array which originated in JS. The functionality of this method allows us to redesign how our array represent data in a way that diverse the data in either a minor or major manner. For instance, this function has been applied to the StackBarChart class, in which the array is called and re-mapped into a new variable. I can then pull these freshly crafted data and applied to all necessarily functions and display it to view.

```
this.maxValue = max(this.listValues);
```

Figure 8 variable containing max function

The max function of p5 is used to return the largest number and store it in the maxValue variable (fig 8).

```
scaleData(_num) {
    let newValue = map(_num, 0, this.maxValue, 0, this.chartHeight);
    return newValue;
}
```

Figure 9 Scaling data to a new range

This variable is then used in the scaleData() method (fig 9) to re-scale the data value to the height of the chart. The result (fig 10) as shown below.

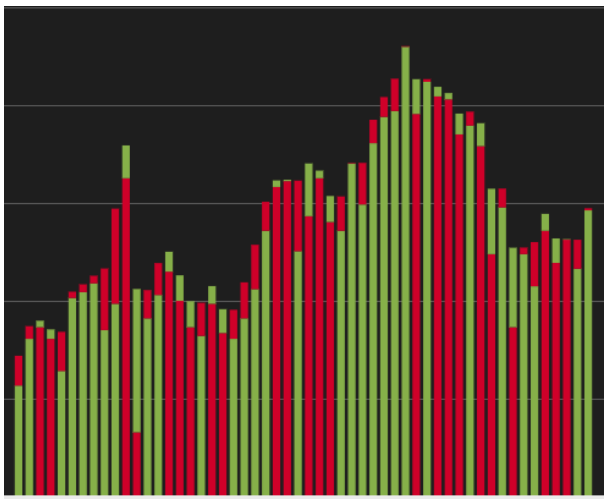

Figure 10 The result of scaleData() method

ArrayName.reduce()

Likewise, the reduce function is a method derived from vanilla JS. This callback function allows us to derive each element in the array that has a unique condition such that, in my case, each data value encapsulated in the objects could be extracted and add to get the total value for each object in the array or to do some sort of calculations before returning the new values.

```
let data2 = [{
    label: "USA",
    categories:["Apples","Pears","Bananas"],
    value:[20,40,10]
},
{
    label: "GERMANY",
    categories:["Apples","Pears","Bananas"],
    value:[60,30,10]
},
{
    label: "IE",
    categories:["Apples","Pears","Bananas"],
    value:[65,70,20]
}];
```

Figure 11 Declaring a variable to store the data set

The function is encapsulated within the For Loop to iterate through the data array (fig 11), so that the object variable in the class can be called which in this case is the values property. The function produces outputs of previous and current array values.

```
for (let i = 0; i < this.data.length; i++) {
        this.data[i].value.reduce(
            (prevValue, curValue) =>
            this.tempArray[i] = prevValue + curValue
            );
```

Figure 12 Array.Reduce method

These values are then get added together and store in a new array called tempArray (fig 12). The results has been logged to check for possible errors (fig 13).

```
70                                          stackedPercentBar.js:73
100                                         stackedPercentBar.js:73
155                                         stackedPercentBar.js:73
```

Figure 13 The console.log of the total values

The array would be further utilised in the scaleData() method which takes in two parameter(param), in which the _num param takes the current value inside each object in the each array index and map it to the chartHeight while the old max range i.e. _array will be the added total value implemented using the reduce method (fig 14).

```
scaleData(_num, _array) {
    let newValue = map(_num, 0, _array, 0, this.chartHeight);
    return newValue;
}
```

Figure 14 scaleData revised to work with stacked bar chart

```
rect(this.barWidth * i + (this.barSpacing*i), 0, this.barWidth, this.scaleData(-this.data[i].value[j], this.tempArray[i]));
translate(0, this.scaleData(-this.data[i].value[j], this.tempArray[i]));
```

Figure 15 scaleData method to used to create the bars

This is the result of the calculation and coding using the array's reduce function (fig 16).

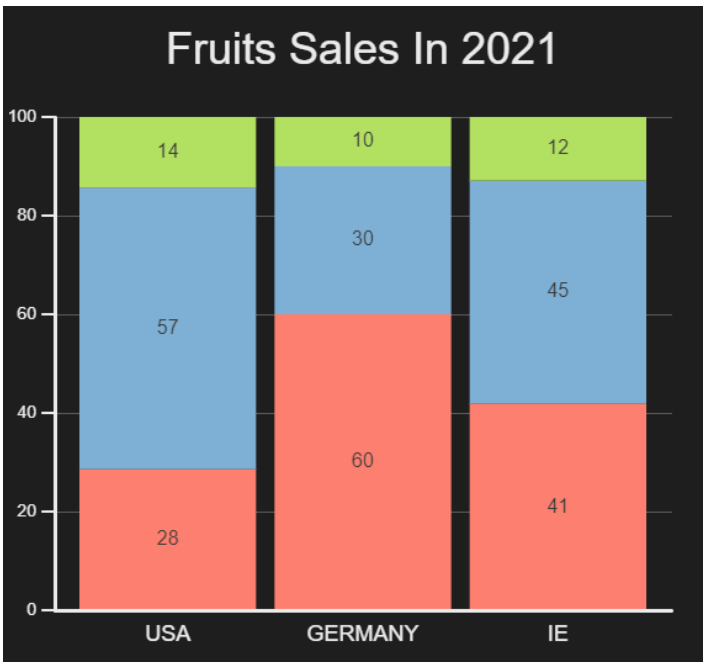

Figure 16 Result of the calculations

Github Repository Link:

https://github.com/Kittitat-13amrung/CC2

References

- The Coding Train. (2017, October 16). *7.4: Mouse Interaction with Objects - p5.js Tutorial* [Video]. YouTube. https://www.youtube.com/watch?v=TaN5At5RWH8
- The Processing Foundation. (2000). *reference | p5.js*. P5 JS. Retrieved March 6, 2022, from https://p5js.org/reference/
- Yahoo Finance. (2022, March 6). *Ethereum Currency Data*. Retrieved March 6, 2022, from https://finance.yahoo.com/quote/ETH-USD?p=ETH-USD
- Mozilla. (2022, February 18). *Array.prototype.reduce() - JavaScript*. Mozilla.Org. Retrieved March 6, 2022, from https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/Reduce
- *How to read candlestick charts*. (n.d.). Coinbase. Retrieved March 6, 2022, from https://www.coinbase.com/learn/tips-and-tutorials/how-to-read-candlestick-charts