Class Diagram

Palette Class - Descriptions, Methods' Preconditions & Postconditions

CD-1A

GeneratePalette()

Preconditions
+ None

Postconditions
+ Either a new color palette is generated or remains the same and returned to the caller

+ User will be notified if a new color cannot be generated due to network issues

AdjustPalette()

Preconditions
+ A palette with no saturation (or grayscaled) cannot be adjusted

+ A set of colors must exist

Postconditions
+ The existing palette colors' saturation and brightness are adjusted or remain the same and returned to the caller

+ User will be notified if the palette's colors cannot be adjusted if it contains any grayscaled values

CopyPalette()

Preconditions
+ A set of colors must exist

Postconditions
+ Copies the palette colors, in hexadecimal format, to the clipboard.

+ User will be notified if the palette's colors they just copied

SetColors(newColors: any[ ])

Preconditions
+ The array of colors passed in must be in TinyColor format or a format that is accepted by TinyColor, such as string Hexadecimal format

Postconditions
+ Formats the array of colors into TinyColor objects and sets the palette's colors property with the converted TinyColor array when a class object is instantiated.

Palette

+ colors: TinyColor [ ]

+ colorPairs: ColorPair [ ]

+ _colorGeneratorService: ColorMindApiService

+ _contrastColorService: ContrastCheckerApiService

+ GeneratePalette(): Palette

+ AdjustPalette(): Palette

+ CopyPalette(): void

+ SetColors(newColors: any[ ]): void

+ CreateColorPairs(colors: TinyColor[ ]): void

+ UpdateColorPairsRatings(colorPairs: ColorPair[ ]): void

+ RandomRange(min: int, max: int): int

+ AdjustColorBrightnessSaturation(color: TinyColor, satMin: int, satMax: int, brightMin: int, brightMax: int): void

CreateColorPairs(colors: TinyColor[ ])

Preconditions
+ A set of colors must be provided
+ color array must be a collection of TinyColor objects

Postconditions
+ Create an array of color pair objects from the color array , estimate each color pair contrast ratings, and set the palette's colorPairs property when a class object is instantiated

UpdateColorPairsRatings(colorPairs: ColorPair[ ])

Preconditions
+ An array of ColorPair objects must exist and be passed in

Postconditions
+ Update the array of color pair objects with contrast ratings retrieved by the ContrastCheckerAPIService class.

RandomRange(min: int, max: int)

Preconditions
+ A set of minimum and maximum numbers must be provided

Postconditions
+ Returns a random number within the range of the specified minimum and maximum numbers

AdjustColorBrightnessSaturation(color: TinyColor, satMin: int, satMax: int, brightMin: int, brightMax: int)

Preconditions
+ A TinyColor object must be provided

+ A set of minimum and maximum numbers must be provided for saturation

+ A set of minimum and maximum numbers must be provided for brightness

Postconditions
+ Adjusts the saturation and brightness values of the TinyColor objects within the specified ranges.

Additional Description: Palette Class / Object

The global Palette Class/Object is a critical piece of the application. It provides all the color and paired color data needed to hydrate the content and modify the HTML element styles of various components and pages within the application. Its properties are used as arguments for multiple methods and functions in the application, and its methods are passed to close child components as props.

A global Palette object is instantiated at the start of the application and stored in as a state using React's UseState hook. The global Palette state is located in the top parent component and accessible to other child components through React's Context hook.

A palette context and context provider is created for the application at the top level, and the palette state tuple, [palette, SetPalette], is assigned as the context value property.

The Palette object in the state can be set/replaced anywhere in the application using the palette context and setPalette() method it holds, so long as the child component consumes the context. And any child components that consume the context will subscribe to its changes.

The Palette Class/Object itself cannot update the global state from within because calling hooks are forbidden in classes, so various Palette methods return a new Palette object to the caller so that a new palette state can be set globally.