

# **ColourPaletteExtractor**

*Release 0.5.5*

**Tim Churchfield**

Aug 01, 2021



<b>1</b>	<b>colourpaletteextractor</b>	<b>1</b>
1.1	colourpaletteextractor package . . . . .	1
	<b>Python Module Index</b>	<b>23</b>
	<b>Index</b>	<b>25</b>



---

## colourpaletteextractor

---

### 1.1 colourpaletteextractor package

#### 1.1.1 Subpackages

##### colourpaletteextractor.controller package

###### Submodules

*colourpaletteextractor.controller.controller module*

###### class

`colourpaletteextractor.controller.controller.ColourPaletteExtractorController`  
( *model*: `colourpaletteextractor.model.model.ColourPaletteExtractorModel`, *view*: `colourpaletteextractor.view-mainview.MainView` )

Bases: `PySide2.QtCore.QRunnable`

ColourPaletteExtractor Controller.

Used to connect the ColourPaletteExtractor GUI signals with the appropriate slot to be able to manipulate the associated model.

**Parameters**

- **model** (`ColourPaletteExtractorModel`) – The main model of ColourPaletteExtractor.
- **view** (`MainView`) – The main window of ColourPaletteExtractor.

**current\_tab\_changed** (*i*: `int` )

Update the current tab index and update the view with the tab's properties.

In most cases,  $i \geq 0$ , however a value of  $i = -2$  or  $-3$  is also valid for performing a 'dummy' tab change to update the current view shown to the user. A value of  $-1$  will lead to the creation of the default tab (the quick start guide).

**Parameters** **i** (`int`) – Index of the current tab.

**Raises**      **ValueError** – If the value of *i* is less than  $-3$ .

*colourpaletteextractor.controller.worker module*

**class** `colourpaletteextractor.controller.worker.Worker` ( *fn*, *function\_type*: `str`, *tab*: `colourpaletteextractor.view.tabview.NewTab`, \**args*, \*\**kwargs* )

Bases: `PySide2.QtCore.QRunnable`

Worker thread used to generate the colour palette or report for an image.

Inherits from `QRunnable` to handler worker thread setup, signals and wrap-up.

Adapted from: [ref](#)

Accessed: 01/08/21

**Parameters**

- **fn** – The function or method to be run as a new thread (generating an image's

colour palette or its colour palette report.

- **tab** (*NewTab*) – `tabview.NewTab` object associated with the image to be processed.
- **function\_type** (*str*) – The action to be run. This can either be 'colour palette' or 'report'.
- **\*args** – Arguments to pass to the callback function
- **\*kwargs** – Keywords to pass to the callback function

Attributes:

**Parameters** **progress\_callback** – The function callback to run on this worker thread. Supplied args and kwargs will be passed through to the runner.

**Raises** **ValueError** – If the provided `function_type` is invalid.

**run ( )**

Initialise the runner function with passed args, kwargs.

**class** `colourpaletteextractor.controller.worker.WorkerSignals`

Bases: `PySide2.QtCore.QObject`

Specify the signals available from a running `Worker` thread.

Adapted from: [ref](#)

Accessed: 01/08/21

Supported signals are:

**error**

Tuple (`exc_type`, `value`, `traceback.format_exc()`).

**finished**

Integer emitted upon finishing.

When generating a colour palette, the value is -2. When generating a report, the value is -3. This is used to reload the tab displaying the image with the correct settings and colour palette.

**progress**

`NewTab` object for which the GUI is to be updated for and the percentage complete for the current task.

The current task is either generating the colour palette for an image or generating the colour palette report for the image.

**result**

Object data returned from processing, anything - NOT IN USE.

**staticMetaObject** = `<PySide2.QtCore.QMetaObject object>`

*Module contents*

**colourpaletteextractor.examples package**

*Submodules*

*colourpaletteextractor.examples.generatecolourpaletteexample module*

Contains an example script demonstrating how to generate the colour palette of a sample image.

*Module contents*

**colourpaletteextractor.model package**

*Subpackages*

*colourpaletteextractor.model.algorithms package*

*Submodules*

*colourpaletteextractor.model.algorithms.cielabcube module*

**class** colourpaletteextractor.model.algorithms.cielabcube.CielabCube (   
 *l\_star\_coord: int, a\_star\_coord: int, b\_star\_coord: int* )

Bases: object

A cube representing a fixed region in the CIELAB colour space.

The cube is used to hold pixels in an image that exist within the cube's region of the CIELAB colour space. The input parameters do not refer to the actual L\*, a\* and b\* values, but depend on the *CUBE\_SIZE* specified by the colour palette algorithm (in particular, any variant on the [Nieves 2020](#) algorithm).

In the case of the `nieves2020.Nieves2020CentredCubes` algorithm, the coordinates refer to the centre of the cube. For the `nieves2020.Nieves2020OffsetCubes` algorithm, the coordinates refer to the corner of the cube closest to the origin.

**Parameters**

- **l\_star\_coord** (*int*) – Perceptual lightness cube coordinate
- **a\_star\_coord** (*int*) – Green-red cube coordinate
- **b\_star\_coord** (*int*) – Blue-yellow cube coordinate

**add\_pixel\_to\_cube** ( *pixel: numpy.array, c\_star: numpy.float64* ) → None

Assign a pixel to the cube.

**Parameters**

- **pixel** (*np.array*) – The pixel as a [L\*,a\*,b\*] triplet.
- **c\_star** (*np.float64*) – The C\* (chroma, relative saturation) value for the pixel.

**property c\_stars: list**

The C\* (chroma, relative saturation) values for all of the pixels in the cube.

$C^{\{*\}} = \sqrt{a^{\{*\}}^2 + b^{\{*\}}^2}$

**Returns** (*list[**np.float64**]*) – The list of C\* values for all pixels in the cube.

**calculate\_mean\_colour** ( ) → None

Calculate the mean colour of the pixels in the cube.

Nothing is calculated if the number of pixels in the cube is equal to 0.

**property coordinates: numpy.array**

The coordinates of the cube ([L\*, a\*, b\*]).

In the case of the `nieves2020.Nieves2020CentredCubes` algorithm, the coordinates refer to the centre of the cube. For the `nieves2020.Nieves2020OffsetCubes` algorithm, the coordinates refer to the corner of the cube closest to the origin.

**Returns** (*np.array*) – The cube's coordinates.

**get\_c\_star\_percentile\_value** ( *percentile: float* ) → Union[int, numpy.percentile]

Returns the C\* value for the given percentile based on the pixels in the cube.

**Parameters** **percentile** (*float*) – The percentile to calculate the C\* value for.

**Returns** (*Union[int, **np.percentile**]*) – The C\* value for the chosen percentile. If no pixels are found, the return value is 0.

**get\_l\_star\_percentile\_value** ( *percentile: float* ) → Union[int, numpy.percentile]

Returns the L\* value for the given percentile based on the pixels in the cube.

**Parameters** **percentile** (*float*) – The percentile to calculate the L\* value for.

**Returns** (*Union[int, **np.percentile**]*) – The L\* value for the chosen percentile. If no pixels are found, the return value is 0.

**increment\_pixel\_count\_after\_reassignment** ( ) → None  
 Increase the number of pixels with this cube's mean colour by one.

**property l\_stars: numpy.array**  
 The L\* values for all pixels in the cube.  
**Returns (np.array)** – Array of L\* values for all pixels in the cube.

**property mean\_colour: numpy.array**  
 The mean colour of the pixels in the cube.  
**Returns (np.array)** – The mean colour of the cube as a [L\*,a\*,b\*] triplet.

**property pixel\_count\_after\_reassignment: int**  
 The number of pixels in the recoloured image with this cube's mean colour.  
**Returns (int)** – The number of pixels with the cube's mean colour.

**property pixels: list**  
 The list of pixels ([L\*, a\*, b\*] triplets) in the cube.  
**Returns (list[np.array])** – The list of pixels in the cube.

**property relevant: bool**  
 The relevancy status of the cube.  
**Returns (bool)** – True if the cube is a relevant cube. Otherwise False.

`colourpaletteextractor.model.algorithms.cielabcube.get_relative_frequencies`  
 ( *relevant\_cubes: list, total\_pixels: int* ) → list

Calculate the relative frequency of each colour (relevant colour) in the recoloured image.

**Parameters**

- **relevant\_cubes** (*list[CielabCube]*) – List of relevant `CielabCube` objects.
- **total\_pixels** (*int*) – The total number of pixels in the image.

**Returns** (*list[float]*) – The list of relative frequencies for each relevant cube.

*colourpaletteextractor.model.algorithms.nieves2020 module*

**class** `colourpaletteextractor.model.algorithms.nieves2020.Nieves2020` ( *name, url* )  
 Bases:  
`colourpaletteextractor.model.algorithms.palettealgorithm.PaletteAlgorithm`,  
`abc.ABC`

Abstract class representing an algorithm to extract the colour palette from an image.

Based on the algorithm proposed by Nieves et al. (2020); see *algorithm* for more information.

**CUBE\_SIZE = 20**  
 Default delta-E\* length of cube side (units).

**C\_STAR\_PERCENTILE = 50**  
 x-th percentile of C\* (50%) for secondary relevancy requirements.

**MIN\_L\_STAR = 80**  
 Minimum L\* value for secondary relevancy requirements (units).

**SECONDARY\_THRESHOLD = 0.00375**  
 Minimum secondary pixel count (%) for secondary relevancy requirements.

**THRESHOLD = 0.03**  
 Minimum threshold colour in each cube (0.03 = 3%) for primary relevancy requirements.



**generate\_colour\_palette** ( *image: numpy.array* ) → tuple

Generate the colour palette and the recoloured image of the provided image.

**Parameters** **image** (*np.array*) – The image for which the colour palette is to be generated (in sRGB colour space).

**Returns**

- (**np.array**) – The recoloured image using only the colours in the colour palette.
- (**list[*np.array*]**) – The list of colours ([R,G,B] triplets) in the image's colour palette.
- (**list[*float*]**) – The relative frequencies of the colours in the recoloured image.

**class**

`colourpaletteextractor.model.algorithms.nieves2020.Nieves2020CentredCubes`

Bases: `colourpaletteextractor.model.algorithms.nieves2020.Nieves2020`

Subclass of `Nieves2020` with the cube coordinates corresponding to the centre of the cube.

As a result, there is only one cube that touches the origin in the CIELAB colour space (is in fact centred on the origin).

**NAME** = 'Nieves, Gomez-Robledo, Chen and Romero (2020) - Cube centred on CIELAB origin'  
Name of the algorithm.

**URL** = '<https://doi.org/10.1364/AO.378659>'  
Link to more information about the algorithm.

**class**

`colourpaletteextractor.model.algorithms.nieves2020.Nieves2020OffsetCubes`

Bases: `colourpaletteextractor.model.algorithms.nieves2020.Nieves2020`

Subclass of `Nieves2020` with the cube coordinates corresponding to the cube's corner closest the origin.

As a result, there are eight cube that touch the origin in the CIELAB colour space.

**NAME** = 'Nieves, Gomez-Robledo, Chen and Romero (2020) - Cube corners at CIELAB origin'  
Name of the algorithm.

**URL** = '<https://doi.org/10.1364/AO.378659>'  
Link to more information about the algorithm.

`colourpaletteextractor.model.algorithms.nieves2020.convert_lab_2_rgb` ( *image: numpy.array* ) → *numpy.array*

Convert an image from the CIELAB colour space to the sRGB colour space.

After conversion, the image is scaled to 8-bit per colour channel (24-bit image).

Illuminant = D65 (name of the illuminant) Observer = 2 (aperture angle of observer)

**Parameters** **image** (*np.array*) – The image in the CIELAB colour space.

**Returns** (**np.array**) – The image in the sRGB colour space.

`colourpaletteextractor.model.algorithms.nieves2020.convert_rgb_2_lab` ( *image: numpy.array* ) → *numpy.array*

Convert an image from the sRGB colour space to the CIELAB colour space.

The alpha channel of the image (RGBA) is removed if present.

Illuminant = D65 (name of the illuminant) Observer = 2 (aperture angle of observer)

**Parameters** **image** (*np.array*) – The image in the sRGB colour space.

**Returns** (`np.array`) – The image in the CIELAB colour space.

`colourpaletteextractor.model.algorithms.nieves2020.get_c_stars ( lab: numpy.array ) → numpy.array`

Get the matrix of C\* (chroma) values for each pixel in the image.

$$C^{\{*\}} = \sqrt{a^{\{*\}}^2 + b^{\{*\}}^2}$$

**Parameters** `lab` (`np.array`) – The image in the CIELAB colour space.

**Returns** (`np.array`) – Array of C\* values corresponding to each pixel in the image.

*colourpaletteextractor.model.algorithms.palettealgorithm module*

**class**

`colourpaletteextractor.model.algorithms.palettealgorithm.PaletteAlgorithm ( name: str, url: str )`

Bases: `abc.ABC`

Abstract class representing an algorithm used to obtain a colour palette from an image.

**Parameters**

- **name** (`str`) – Name of the algorithm
- **url** (`str`) – Link to a description of the algorithm

**property** `continue_thread: bool`

Get the execution status of the algorithm.

A value of *false* would indicate that the algorithm should return without generation a colour palette when it next checks its execution status.

**Returns** `bool` – The execution status of the algorithm

**abstract** `generate_colour_palette ( image: numpy.array ) → tuple`

Generate the colour palette for the given image.

Analyses the given image to obtain its colour palette. Returns the recoloured image using only the colours found in the colour palette, the colour palette of the image and finally the relative frequencies of each of those colours in the recoloured image.

**Parameters** `image` (`np.array`) – A 3D array representing an image. It is assumed that the input image is

**Returns**

- **recoloured\_image** (`np.array`) – The recoloured image using only the colours found in the colour palette
- **colour\_palette** (`list`) – The list of colours (sRGB 8-bit values) in the colour palette
- **relative\_frequencies** (`list`) – The relative frequencies of each colour in the colour palette in the recoloured image

---

**Note** It is assumed that the input image has been encoded in the sRGB colour space.

---

**property** `name: str`

Get the name of the algorithm.

**Returns** (`str`) – The name of the algorithm

**set\_progress\_callback** ( `progress_callback: PySide2.QtCore.SignalInstance, tab: colourpaletteextractor.view.tabview.NewTab, image_data` ) → `None`

Set the signal function called by the algorithm at regular intervals to update the GUI thread.

**Parameters**

- **progress\_callback** (`QtCore.SignalInstance`) – Signal that when emitted, is used to update the GUI.
- **tab** (`NewTab`) – The tab associated with the image being analysed (see

`generate_colour_palette()`.

- **image\_data** (*ImageData*) – *ImageData* object that holds the image being analysed.

**property url:** str

Get the link to the description of the algorithm.

**Returns** (str) – The link to the description of the algorithm

`colourpaletteextractor.model.algorithms.palettealgorithm.get_implemented_algorithms()`

Recursively finds all subclasses of the `PaletteAlgorithm` class.

Like Python's `__class__.__subclasses__()`, but recursive. Returns a list containing all subclasses of `PaletteAlgorithm`.

Adapted from: [ref](#)

Accessed: 15/07/2021

**Returns:**

[object]: List of all non-abstract subclasses of `PaletteAlgorithm`

*Module contents*

*Submodules*

*colourpaletteextractor.model.generatereport module*

**class** `colourpaletteextractor.model.generatereport.ColourPaletteReport` (*image\_data: colourpaletteextractor.model.imagedata.ImageData*)

Bases: `fpdf.fpdf.FPDF`, `fpdf.html.HTMLMixin`

A modified FPDF object to fit the requirements for generating a PDF colour palette report.

**Parameters** **image\_data** (*ImageData*) – The *ImageData* object holding the image's data (the original image, the recoloured image, and the colour palette).

**A4\_HEIGHT** = 297

The height of an A4 sheet of paper (mm).

**A4\_WIDTH** = 210

The width of an A4 sheet of paper (mm).

**IMAGE\_START\_POSITION** = 30

The standard left indentation when placing an image in the PDF report (mm).

**IMAGE\_WIDTH** = 150

The standard width of images in the PDF report (mm).

**MARGIN** = 10

The size of the margins to be used in the PDF report (mm).

**MAX\_IMAGE\_HEIGHT** = 257

The standard maximum height of images in the report (mm).

**footer** ( ) → None

Set the footer used in the PDF report.

**header** ( ) → None

Set the header used in the PDF report.

**class** colourpaletteextractor.model.generatereport.**ReportGenerator** ( *tab: colourpaletteextractor.view.tabview.NewTab, image\_data: colourpaletteextractor.model.imagedata.ImageData, progress\_callback: PySide2.QtCore.SignalInstance* )

Bases: object

Class used to create, populate a `ColourPaletteReport` object and save the resulting PDF to disk.

**Parameters**

- **tab** (*NewTab*) – The tab associated with the image to be analysed.
- **image\_data** (*ImageData*) – The `ImageData` object holding the image’s data (the original image, the recoloured image, and the colour palette).
- **progress\_callback** (*QtCore.SignalInstance*) – Signal that when emitted, is used to update the GUI.

**create\_report** ( ) → Optional[`colourpaletteextractor.model.generatereport.ColourPaletteReport`]

Create a `ColourPaletteReport` object representing the PDF colour palette report.

**Returns** (Union[`ColourPaletteReport`, None]) – None if the `ColourPaletteReport` object was not properly generated, otherwise returns the populated `ColourPaletteReport` object.

**save\_report** ( *pdf: colourpaletteextractor.model.generatereport.ColourPaletteReport* ) → None

save the `ColourPaletteReport` object representing the PDF colour palette report to disk.

**Parameters** **pdf** (*ColourPaletteReport*) – The `ColourPaletteReport` object to be saved as a PDF to disk..

colourpaletteextractor.model.generatereport.**generate\_report** ( *tab: colourpaletteextractor.view.tabview.NewTab, image\_data: colourpaletteextractor.model.imagedata.ImageData, progress\_callback: PySide2.QtCore.SignalInstance* ) → None

Generate a colour palette report for an image.

**Parameters**

- **tab** (*NewTab*) – The tab associated with the image to be analysed.
- **image\_data** (*ImageData*) – The `ImageData` object holding the image’s data (the original image, the recoloured image, and the colour palette).
- **progress\_callback** (*QtCore.SignalInstance*) – Signal that when emitted, is used to update the GUI.

**Raises** **ValueError** – If the provided `ImageData` object does not have a recoloured image or has no colours in its colour palette.

*colourpaletteextractor.model.imagedata module*

**class** colourpaletteextractor.model.imagedata.**ImageData** ( *file\_name\_and\_path: str* )

Bases: object

Object to hold the data associated with an image to be analysed.

Stores the original image, its colour palette, the recoloured image, the relative frequency of each colour in the recoloured image, the algorithm used to generate the colour palette and the execution status of the thread used to generate the colour palette.

**Parameters** **file\_name\_and\_path** (*str*) – Path to the image to be added.

**Raises** **ValueError** – If the `file_name_and_path` argument is None.

**property algorithm\_used:** type

The algorithm used to generate the image’s colour palette.

**Returns** (type[`palettealgorithm.PaletteAlgorithm`]) – The class name of the colour palette extraction algorithm.

**property colour\_palette: list**

The list of colours in the image's colour palette.

**Returns (list[np.array])** – The list of colours ([R,G,B] triplets) in the colour palette.

**property colour\_palette\_relative\_frequency: list**

The relative frequencies of each colour in the colour palette in the recoloured image.

The order of the relative frequencies matches the order of the colours in the colour palette.

**Returns (list[float])** – The list of relative frequencies of the colour palette.

**property continue\_thread: bool**

Specify if the thread for generating the colour palette or the report should be cancelled.

**Returns (bool)** – True if the thread should be continued. Otherwise False.

**property extension: str**

The file extension of the original image.

**Returns (str)** – The file extension of the original image.

**property file\_name\_and\_path: str**

The file path to the original image.

**Returns (str)** – File path to the original image.

**static get\_image\_as\_q\_image ( image: numpy.array ) → PySide2.QtGui.QImage**

Convert a Numpy array representation of an image to a QImage.

**Parameters image (np.array)** – An image represented by a Numpy array.

**Returns (QImage)** – The image converted to a QImage.

**Raises ValueError** – If the provided image is not a greyscale, rGB or RGBA image (1, 3, or 4 colour channels).

**property image: numpy.array**

The original image, represented as a 2 or 3-D Numpy array.

**Returns (np.array)** – The original image as a Numpy array.

**property name: str**

The name of the image, without its file extension.

**Returns (str)** – The image file name, without its extension.

**property recoloured\_image**

The recoloured image, represented as a 3-D Numpy array.

**Returns (np.array)** – The recoloured image as a Numpy array.

**sort\_colour\_palette ( reverse: bool = True ) → None**

Sort the colour palette by their relative frequencies in the recoloured image.

**Parameters reverse (bool)** – If True, the colour palette is sorted from largest relative frequency to the smallest. If False, the order is smallest to largest. The default is True.

*colourpaletteextractor.model.model module*

**class colourpaletteextractor.model.model.ColourPaletteExtractorModel**

Bases: object

ColourPaletteExtractor Model.

Used as the model component of the ColourPaletteExtractor application.

**DEFAULT\_ALGORITHM**

alias of  
`colourpaletteextractor.model.algorithms.nieves2020.Nieves2020CentredCubes`

**DEFAULT\_HEIGHT: int = 894**

Default height of the ColourPaletteExtraction application.  
 Size chosen to show the Quick Start Guide image without the need of scrollbars.

**DEFAULT\_USER\_DIRECTORY: str = '/Users/tim/Documents/ColourPaletteExtractor/Output'**

The default user output directory for colour palette reports.

**DEFAULT\_USE\_USER\_DIRECTORY: bool = False**

Specify by default whether a user's output directory should be used for saving the colour palette report to.

**DEFAULT\_WIDTH: int = 1523**

Default width of the ColourPaletteExtraction application.  
 Size chosen to show the Quick Start Guide image without the need of scrollbars.

**SUPPORTED\_IMAGE\_TYPES: set = {'jpeg', 'jpg', 'png'}**

The set of supported image extensions.

**property active\_thread\_counter: int**

The number of active threads still running as part of a batch operation.

**Returns (int)** – The number of active threads still running.

**add\_image (file\_name\_and\_path: str) → tuple**

Given the path to an image, create a new `ImageData` object and return it and its ID key.

**Parameters file\_name\_and\_path (str)** – Path to the image.

**Returns**

- **(str)** – The dictionary key ('Tab\_xx') for the new `ImageData` object in the `image_data_id_dictionary`.
- **(ImageData)** – The new `ImageData` object for holding information about the image (e.g., the colour palette, the recoloured image etc.)

**Raises**     **KeyError** – If the generated dictionary key already exists in the model's dictionary of `ImageData` objects     (`image_data_id_dictionary`).

**change\_output\_directory (use\_user\_dir: bool, new\_user\_directory: str) → None**

Change the output directory for colour palette reports in the `ColourPaletteExtractor.ini` settings file.

**Parameters**

- **use\_user\_dir (bool)** – True if the user-selected output directory is to be used. If False, use default temporary output directory.
- **new\_user\_directory (str)** – The path to the new user-selected output directory

**close\_temporary\_directory ( ) → None**

Delete the temporary output directory associated with the instance of the application.

**generate\_palette (image\_data\_id: str, tab: Optional[colourpaletteextractor.view.tabview.NewTab] = None, progress\_callback=None, algorithm: Optional[type] = None) → None**

Generate the colour palette for the image in the `ImageData` object with the given `image_data_id` ID.

The recoloured image, colour palette and relative frequencies of each colour are added to the `ImageData` object with the `image_data_id` dictionary key.

- Parameters**
- **image\_data\_id** (*str*) – The dictionary key/ID ('Tab\_xx') for the ImageData object in the `image_data_id_dictionary` for which the colour palette of its associated image is to be generated for.
  - **tab** (*NewTab*) – The NewTab linked to the image that is to have its colour palette generated.
  - **progress\_callback** (*QtCore.SignalInstance*) – Signal that when emitted, is used to update the GUI.
  - **algorithm** (*type[PaletteAlgorithm]*) – The algorithm class to be used to generate the colour palette.

**get\_image\_data** (*image\_data\_id: str*) → `colourpaletteextractor.model.imagedata.ImageData`

Returns the ImageData object with the given ID/key in the `image_data_id_dictionary`.

**Parameters** **image\_data\_id** (*str*) – The dictionary key/ID ('Tab\_xx') for the ImageData object in the `image_data_id_dictionary` that should be returned.

**Returns** (**ImageData**) – ImageData object with the given ID/key.

**property image\_data\_id\_dictionary: dict**

The dictionary storing the ImageData objects for the images currently open.

**Returns** (**dict**) – dictionary storing the ImageData objects for the images currently open.

**remove\_image\_data** (*image\_data\_id: str*) → None

Remove ImageData object from the dictionary of images (`image_data_id_dictionary`) by its key.

**Parameters** **image\_data\_id** (*str*) – The dictionary key ('Tab\_xx') for the ImageData object in the `image_data_id_dictionary` that should be removed.

**set\_algorithm** (*algorithm\_class: type = <class 'colourpaletteextractor.model.algorithm-s.nieves2020.Nieves2020CentredCubes'>*) → None

Set the algorithm used to generate the colour palette of an image.

If no `algorithm_class_name` is provided, the `DEFAULT_ALGORITHM` is used.

**Parameters** **algorithm\_class** (*type[PaletteAlgorithm]*) – The algorithm class.

**write\_default\_settings** () → None

Write the default settings to the ColourPaletteExtractor.ini settings file.

`colourpaletteextractor.model.model.generate_colour_palette_from_image` (*path\_to\_file: str, algorithm: Optional[type] = None*) → tuple

Generate the colour palette for the given images using the specified colour palette extraction algorithm.

An example algorithm would be `nieves2020.Nieves2020CentredCubes`

- Parameters**
- **path\_to\_file** (*str*) – Path to the image to be analysed.
  - **algorithm** (*type[PaletteAlgorithm]*) – The Python class of the the colour palette extraction algorithm.

- Returns**
- (**np.ndarray**) – The recoloured image using just the colours in the colour palette.
  - (**list[np.ndarray]**) – The list of colours ([R,G,B] triplets) in the colour palette.
  - (**list[float]**) – The relative frequencies of the colours in the colour palette in the recoloured image.



`colourpaletteextractor.model.model.get_settings ( )` → PySide2.QtCore.QSettings

Get the settings file for the ColourPaletteExtraction application.

**Returns (QSettings)** – The settings for the ColourPaletteExtraction application.

*Module contents*

**colourpaletteextractor.tests package**

*Subpackages*

*colourpaletteextractor.tests.helpers package*

*Submodules*

*colourpaletteextractor.tests.helpers.helperfunctions module*

`colourpaletteextractor.tests.helpers.helperfunctions.get_image ( path_to_image: str )`

Returns the image found at the given path.

**Parameters** `path_to_image (str)` – Path to the image to be imported.

**Returns** `(np.array)` – Image represented as a 3D array

*Module contents*

*Submodules*

*colourpaletteextractor.tests.nieves2020\_test module*

`colourpaletteextractor.tests.nieves2020_test.test_closest_relevant_colour_used_to_recol ( )`

`colourpaletteextractor.tests.nieves2020_test.test_cube_colour_must_occur_more_than_thre ( )`

`colourpaletteextractor.tests.nieves2020_test.test_cube_colour_must_occur_more_than_thre ( )`

`colourpaletteextractor.tests.nieves2020_test.test_low_a_b_colour_does_not_meet_secondar ( )`

`colourpaletteextractor.tests.nieves2020_test.test_low_a_b_colour_does_not_meet_secondar ( )`

`colourpaletteextractor.tests.nieves2020_test.test_nieves2020_centred_cubes_constructor ( )`

`colourpaletteextractor.tests.nieves2020_test.test_nieves2020_offset_cubes_constructor ( )`

`colourpaletteextractor.tests.nieves2020_test.test_primary_requirements_1 ( )`

`colourpaletteextractor.tests.nieves2020_test.test_primary_requirements_2 ( )`

`colourpaletteextractor.tests.nieves2020_test.test_primary_requirements_3 ( )`

`colourpaletteextractor.tests.nieves2020_test.test_recoloured_image_of_same_size_1 ( )`

`colourpaletteextractor.tests.nieves2020_test.test_recoloured_image_two_colours_1 ( )`

`colourpaletteextractor.tests.nieves2020_test.test_recoloured_image_two_colours_2 ( )`



```
colourpaletteextractor.tests.nieves2020_test.test_two_colours_in_same_cube_can_meet_sec
( )
```

*Module contents*

**colourpaletteextractor.view package**

*Submodules*

*colourpaletteextractor.view.mainview module*

**class** colourpaletteextractor.view.mainview.**MainView** ( *parent=None* )

Bases: PySide2.QtWidgets.QMainWindow

The main window of the ColourPaletteExtractor application.

**Parameters** **parent** – Parent object of the MainWindow. Defaults to None.

**tabs**

tabbed widget for displaying and managing imported images.

Type QTabWidget

**colour\_palette\_dock**

Type tabview.ColourPaletteDock

**\_close\_request\_action**

Action for closing the application

Type QAction

**open\_action**

Action for opening a new image

Type QAction

**generate\_report\_action**

Action for generating a report for an image

Type QAction

**generate\_all\_report\_action**

Action for generating a report for all images with a colour palette

Type QAction

**generate\_palette\_action**

Action for generating the colour palette for an image

Type QAction

**generate\_all\_palette\_action**

Action for generating the colour palette for all images

Type QAction

**stop\_action**

Action for stopping the report or colour palette being generated for an image

Type QAction

**preferences\_menu\_action**

Action for opening the preferences menu

Type QAction

**show\_help\_action**

Action for showing the quick start guide

Type QAction

**toggle\_recoloured\_image\_action**

Action for toggling between the original and the recoloured image

Type QAction

**zoom\_in\_action**

Action for zooming into an image

Type QAction

**zoom\_out\_action**

Action for zooming out of an image

Type QAction

**about\_menu\_action**

Action for showing the about information widget

Type QAction

**show\_palette\_dock\_action**

Action for showing the colour palette dock

Type QAction

**show\_toolbar\_action**

Action for showing the toolbar

Type QAction

**tools**

(QToolBar): Toolbar for holding QToolButtons used in the GUI

**status**

Status bar for holding hints, the progress bar and the current version of the application

Type [otherviews.StatusBar](#)

**RESOURCES\_DIR = 'resources'**

The name of the directory containing the icons and images used for the GUI.

Type str

**app\_icon = 'app\_icon'**

The name of the file used as the application's icon.

Type str

**closeEvent** ( *event: PySide2.QtGui.QCloseEvent* ) → None

Intercept GUI close event to check if the user wishes to close the GUI.

Parameters **event** (*QtGui.QCloseEvent*) – Close event

**close\_current\_tab** ( *tab\_index: int* ) → int

Close the tab with the given index.

Parameters **tab\_index** (*int*) – The index of the tab to close

Returns (int) – The index of the tab that is now visible after closing the selected tab

**create\_new\_tab** ( *image\_id, image\_data* ) → None

Create a new image tab for the main window.

Parameters • **image\_id** (*str*) – ID of the image to be used for the new tab (e.g., 'Tab\_1')

- **image\_data** (*model.imagedata.ImageData*) – Object containing tab and image properties and state

**default\_new\_tab\_image** = 'images:how-to-dark-mode.png'

The name of the file used as the default new tab (the quick start guide).

Type str

**resources\_path** = '/Users/tim/OneDrive - University of St Andrews/University/MScProject/ColourPaletteExtractor/colourpaletteextractor/view/resources'

The path to the resources used for the GUI.

This will vary depending on whether the code has been compiled into an application or is been run from the command line.

Type str

**show\_file\_dialog\_box** (*supported\_file\_types: set*) → tuple

Show the dialog box for importing images.

**Parameters** *supported\_file\_types* (*set[str]*) – The supported file types (e.g., '.png')

- Returns**
- **list** (*str*) – The list of the absolute paths to the images to be loaded into the application
  - **str** – The filter used when selecting the images to import

**staticMetaObject** = <PySide2.QtCore.QMetaObject object>

*colourpaletteextractor.view.otherviews module*

**class** *colourpaletteextractor.view.otherviews.AboutBox* (*parent=None*)

Bases: PySide2.QtWidgets.QMessageBox

Message box to show the basic information about the application.

**Parameters** *parent* – The parent object of the AboutBox. The default is None.

**staticMetaObject** = <PySide2.QtCore.QMetaObject object>

**class** *colourpaletteextractor.view.otherviews.BatchGenerationProgressWidget*

Bases: PySide2.QtWidgets.QDialog

Custom dialog box shown when multiple colour palette or reports are being generated.

Shows the number of threads to be run and the number of threads completed. Is also has a simple animation attached to it so the user knows that the application has not frozen and is still processing their images.

**label**

Label used to show the number of threads to be run and the number completed.

Type QLabel

**cancel\_batch\_button**

The button used to notify the controller object that the user wishes to cancel the current batch processing.

Type QPushButton

**set\_cancel\_text** ( ) → None

Set the text shown to cancelling to let the user know that any incomplete threads are to be cancelled.

**show\_widget** (*total\_count: int, batch\_type: str*) → None

Reset and show the widget.

- Parameters**
- **total\_count** (*int*) – The total number of threads to be processed.
  - **batch\_type** (*str*) – The text clarifying what task is being carried out as a batch process.

**staticMetaObject** = <PySide2.QtCore.QMetaObject object>

**update\_progress** ( ) → None

Update the batch progress bar by increasing the number of completed threads by one.

**class** colourpaletteextractor.view.otherviews.**ElidedLabel** ( *text='', width=40, parent=None* )

Bases: PySide2.QtWidgets.QLabel

Status bar message label that will become elided if there is not enough space to display the entire message.

Adapted from: [ref1](#) and [ref2](#)

Accessed: 18/07/2021

**Args:**

text (str): The text to be shown in the label. The default is an empty string

width (int): The minimum width of the label. The default is 40.

parent: The parent object of the ElidedLabel. The default is None.

**elided\_text** ( ) → str

Get the elided text shown by the label.

**Returns** (str) – The elided text

**paintEvent** ( *event: PySide2.QtCore.QEvent.Type.Paint* ) → None

Update the text shown by the label on receiving a paint event.

**Parameters** **event** (*QEvent.Type.Paint*) – A paint event

**staticMetaObject** = <PySide2.QtCore.QMetaObject object>

**class** colourpaletteextractor.view.otherviews.**ErrorBox** ( *box\_type: Optional[str] = None, parent=None* )

Bases: PySide2.QtWidgets.QMessageBox

Message box to show warnings and errors.

**Parameters**

- **box\_type** (*str*) – The error box type. Used to customise the icon and main text show.

- **parent** – Parent object of the ErrorBox. Defaults to None.

**header**

The heading of the ErrorBox.

**Type** str

**append\_title** ( *error: Exception* ) → None

Append the title with additional information from an exception.

**Parameters** **error** (*Exception*) – Exception whose error summary message is appended to the title text.

**staticMetaObject** = <PySide2.QtCore.QMetaObject object>

**class** colourpaletteextractor.view.otherviews.**PreferencesWidget** ( *parent=None* )

Bases: PySide2.QtWidgets.QDialog

The dialog box for setting a user's preferences.

Currently, the user can change the algorithm used to generate the colour palette, as well as the output directory for any reports that are generated.

**Parameters** **parent** – The parent object of the PreferencesWidget. The default is None.

**browse\_button**

Button used to open the operating system's file explorer to select a valid output directory.

Type QPushButton

**user\_path\_selector**

Text window used to show the user's currently selected output directory.

Type QLineEdit

**default\_path\_button**

Button used to select the default output directory.

Type QRadioButton

**user\_path\_button**

Button used to select the user's output directory.

Type QRadioButton

**output\_tab**

The output directory settings tab of the preferences dialog box.

Type QWidget

**algorithm\_tab**

The algorithm settings tab of the preferences dialog box.

Type QWidget

**get\_algorithms\_and\_buttons** ( ) → tuple

Get the list of algorithm classes and their associated buttons.

**Returns**     • (list[palettealgorithm.PaletteAlgorithm]) – List of algorithm classes.  
                  • (list[QRadioButton]) – List of buttons associated with the algorithm classes.

**show\_output\_directory\_dialog\_box** ( *current\_path: str* )

Show the dialog box for selecting output directory for reports.

**Parameters** **current\_path** (*str*) – The path to open the system's file explorer to.

**Returns**     (*str*) – Path to the new output directory.

**show\_preferences** ( ) → None

Show the preferences widget.

**staticMetaObject** = <PySide2.QtCore.QMetaObject object>

**update\_preferences** ( ) → None

Update the preferences dialog box with the correct settings.

**class** colourpaletteextractor.view.otherviews.**StatusBar** ( *parent=None* )

Bases: PySide2.QtWidgets.QStatusBar

The status bar at the bottom of the main window.

This holds the current shortcut tip for the given tab, as well as the progress bar for showing the current progress towards generating a report or the image's colour palette.

**Parameters** **parent** – Parent object of the StatusBar. Defaults to None.

**`_status_label`**

Primary status label.

Type `ElidedLabel`

**`_progress_bar`**

Progress bar used to track the progress of generating a colour palette or a report.

Type `QProgressBar`

**`_max_progress`**

Maximum value for the progress bar.

Type `int`

**`_min_progress`**

Minimum value for the progress bar.

Type `int`

**`set_status_bar ( state: int )`** → `None`

Set the state of the status bar elements.

Depending on the state, the primary status label will change to reflect what the application is currently processing.

**Parameters** `state (int)` – The new state of the status bar.

**Raises** `ValueError` – If state is not a valid state.

**`staticMetaObject`** = `<PySide2.QtCore.QMetaObject object>`

**`update_progress_bar ( n: float )`** → `None`

Update the current level of progress for the status bar.

**Parameters** `n (float)` – New level of progress for the progress bar.

**Raises** `ValueError` – If the new progress value exceeds the predefined limits of the progress bar.

*colourpaletteextractor.view.tabview module*

**`class colourpaletteextractor.view.tabview.ColourBox ( parent=None )`**

Bases: `PySide2.QtWidgets.QLabel`

Modified `QLabel` to hold an individual colour in the colour palette.

**Parameters** `parent` – The parent object of the `ColourBox`. The default is `None`.

**`enterEvent ( event: PySide2.QtCore.QEvent )`** → `None`

Intercept an enter event.

In the future, this could be used to trigger the highlighting regions of the image that use this colour in the recoloured image.

**Parameters** `event (QEvent)` – Enter event.

**`leaveEvent ( event: PySide2.QtCore.QEvent )`**

Intercept a leave event.

In the future, this could be used to cancel the highlighting of regions of the image that use this colour in the recoloured image.

**Parameters** `event (QEvent)` – Leave event.

**`staticMetaObject`** = `<PySide2.QtCore.QMetaObject object>`

**`class colourpaletteextractor.view.tabview.ColourPaletteDock ( parent=None )`**

Bases: `PySide2.QtWidgets.QDockWidget`

A modified QDockWidget to hold small images of each colour in an image's colour palette.

**Parameters** **parent** – Parent object of the ColourPaletteDock. Defaults to None.

**add\_colour\_palette** ( *colour\_palette: list, image\_id: str, relative\_frequencies: Optional[list] = None* ) → None

Clear the colour palette dock and add a new image's colour palette to the dock.

- Parameters**
- **colour\_palette** (*list[np.array]*) – List of colours in the colour palette.
  - **image\_id** (*str*) – The ID ('Tab\_xx') associated with a tab and image.
  - **relative\_frequencies** (*list[float]*) – The relative frequencies of each colour in the colour palette in the recoloured image.

**remove\_colour\_palette** ( ) → None

Remove all of the ColourBox labels from the colour palette dock.

Adapted from: [ref](#)

Accessed: 27/07/2021

**staticMetaObject** = <PySide2.QtCore.QMetaObject object>

**class** colourpaletteextractor.view.tabview.**ImageDisplay** ( *image\_data: colourpaletteextractor.model.imagedata.ImageData, parent=None* )

Bases: PySide2.QtWidgets.QLabel

A modified QLabel to display and manipulate the current image.

- Parameters**
- **image\_data** (*imagedata.ImageData*) – The ImageData object that hold the information associated with an image.
  - **parent** – Parent object of the ImageDisplay. Defaults to None.

**event** ( *event: PySide2.QtCore.QEvent* ) → bool

Intercept the QLabel's event if it is a gesture to allow for zooming into and out of the current image.

Also calls the super class' event handler at the end.

**Parameters** **event** (*QEvent*) – An event.

**Returns** (**bool**) – The result from the super class' event handler.

**image\_zoom** ( *mouse\_pos: PySide2.QtCore.QPoint, value: float* ) → None

Zoom into or out of an image at the mouse pointer's current location.

- Parameters**
- **mouse\_pos** (*QtCore.QPoint*) – Current position of the mouse cursor.
  - **value** (*float*) – The degree of magnification of the image.

**staticMetaObject** = <PySide2.QtCore.QMetaObject object>

**update\_image** ( *image: numpy.array* ) → None

Update the image shown by the ImageDisplay.

**Parameters** **image** (*np.array*) – Numpy array representing an image.

**zoom\_factor** = 1.25

The zoom-in factor used when the user zoom's into the image via the zoom-in button.

**zoom\_in** ( *zoom\_factor: float = 1.25* ) → None

Zoom into the current image.

**Parameters** **zoom\_factor** (*float*) – The new magnification factor for the image.

**zoom\_out** ( *zoom\_factor: float = 0.8* ) → None

Zoom out of the current image.

**Parameters** **zoom\_factor** (*float*) – The new magnification factor for the image.

**zoom\_out\_factor = 0.8**

The zoom-out factor used when the user zoom's out of the image via the zoom-out button.

**class** colourpaletteextractor.view.tabview.**NewTab** ( *image\_id: Optional[str] = None, image\_data: Optional[colourpaletteextractor.model.imagedata.ImageData] = None, parent=None* )

Bases: PySide2.QtWidgets.QScrollArea

Modified QScrollArea to display and manipulate an image (via the [ImageDisplay](#) class).

**Parameters**

- **image\_id** (*str*) – The ID ('Tab\_xx') associated with a tab and image.
- **image\_data** (*imagedata.ImageData*) – The ImageData object that hold the information associated with an image.
- **parent** – Parent object of the NewTab Defaults to None.

**image\_display**

ImageDisplay used to show the QPixmap representation of the current image.

**Type** [ImageDisplay](#)

**change\_toggle\_recoloured\_image\_pressed** ( ) → None

Toggle the \_toggle\_recoloured\_image\_pressed attribute between true and false (its opposite).

**property generate\_palette\_available: bool**

The ability to generate the colour palette for the current NewTab object.

**Returns (bool)** – Returns true if the colour palette can be generated. Otherwise false.

**property generate\_report\_available: bool**

The ability to generate the colour palette report for the current NewTab object.

**Returns (bool)** – Returns true if the colour palette report can be generated. Otherwise false.

**get\_slider\_positions** ( ) → PySide2.QtCore.QPointF

Get the grip positions of the horizontal and vertical scrollbars.

**Returns (QPointF)** – The position of the grip for the horizontal and vertical scrollbars.

**property image\_id: str**

The image ID of the images and its data that is linked to the current NewTab object

**Returns (str)** – The ID ('Tab\_xx') associated with a tab and image.

**property progress\_bar\_value: float**

The current level of progress shown by the status bar for the associated NewTab object.

**Returns (float)** – The current level of progress shown by the status bar.

**set\_slider\_positions** ( *x\_position: float, y\_position: float* ) → None

Set the position of the horizontal and vertical scrollbar's grip.

**Parameters**

- **x\_position** (*float*) – Position of the grip for the horizontal scrollbar.
- **y\_position** (*float*) – Position of the grip for the vertical scrollbar.

**staticMetaObject** = <PySide2.QtCore.QMetaObject object>



**property status\_bar\_state: int**

The current status bar state, represented by an integer.

See the `otherviews.StatusBar.set_status_bar()` method for more information.

**Returns (int)** – The current status bar state.

**property toggle\_recoloured\_image\_available: bool**

Stores the availability of the recoloured image (if it available to be displayed or not).

**Returns (bool)** – True if the recoloured image is available. Otherwise false.

**property toggle\_recoloured\_image\_pressed: bool**

The status of the toggle button used to switch between the original image and the recoloured image.

**Returns (bool)** – True if the recoloured image is displayed by the GUI. Otherwise false.

**wheelEvent ( event: PySide2.QtGui.QWheelEvent ) → None**

Intercepts the super class' wheelEvent to allow zooming into and out of an image using the mousewheel.

Also calls the super class' wheelEvent handler at the end.

**Parameters event (QWheelEvent)** – Mousewheel event

**property zoom\_level: float**

The degree of magnification for the currently displayed image.

**Returns (float)** – The degree of magnification for the current image.

*Module contents***1.1.2 Module contents**

- genindex
- modindex
- search



## C

- colourpaletteextractor, 21
  - colourpaletteextractor.controller,
    - 2
  - colourpaletteextractor.controller.controller,
    - 1
  - colourpaletteextractor.controller.worker,
    - 1
  - colourpaletteextractor.examples,
    - 2
  - colourpaletteextractor.examples.generatecolourpaletteexample,
    - 2
  - colourpaletteextractor.model, 12
    - colourpaletteextractor.model.algorithms,
      - 7
    - colourpaletteextractor.model.algorithms.cielabcube,
      - 3
    - colourpaletteextractor.model.algorithms.nieves2020,
      - 4
    - colourpaletteextractor.model.algorithms.palettealgorithm,
      - 6
    - colourpaletteextractor.model.generatereport,
      - 7
    - colourpaletteextractor.model.imagedata,
      - 8
    - colourpaletteextractor.model.model,
      - 9
  - colourpaletteextractor.tests, 13
    - colourpaletteextractor.tests.helpers,
      - 12
    - colourpaletteextractor.tests.helpers.helperfunctions,
      - 12
    - colourpaletteextractor.tests.nieves2020\_test,
      - 12
  - colourpaletteextractor.view, 21
    - colourpaletteextractor.view.mainview,
      - 13
    - colourpaletteextractor.view.otherviews,
      - 15
    - colourpaletteextractor.view.tabview,
      - 18



## Symbols

`_close_request_action` (colourpaletteextractor.view.mainview.MainView attribute), 13

`_max_progress` (colourpaletteextractor.view.otherviews.StatusBar attribute), 18

`_min_progress` (colourpaletteextractor.view.otherviews.StatusBar attribute), 18

`_progress_bar` (colourpaletteextractor.view.otherviews.StatusBar attribute), 18

`_status_label` (colourpaletteextractor.view.otherviews.StatusBar attribute), 18

## A

`A4_HEIGHT` (colourpaletteextractor.model.generatorreport.ColourPaletteReport attribute), 7

`A4_WIDTH` (colourpaletteextractor.model.generatorreport.ColourPaletteReport attribute), 7

`about_menu_action` (colourpaletteextractor.view.mainview.MainView attribute), 14

`AboutBox` (class in colourpaletteextractor.view.otherviews), 15

`active_thread_counter` (colourpaletteextractor.model.model.ColourPaletteExtractorModel property), 10

`add_colour_palette()` (colourpaletteextractor.view.tabview.ColourPaletteDock method), 19

`add_image()` (colourpaletteextractor.model.model.ColourPaletteExtractorModel method), 10

`add_pixel_to_cube()` (colourpaletteextractor.model.algorithms.cielabcube.CielabCube method), 3

`algorithm_tab` (colourpaletteextractor.view.otherviews.PreferencesWidget attribute),

17

`algorithm_used` (colourpaletteextractor.model.imagedata.ImageData property), 8

`app_icon` (colourpaletteextractor.view.mainview.MainView attribute), 14

`append_title()` (colourpaletteextractor.view.otherviews.ErrorBox method), 16

## B

`BatchGenerationProgressWidget` (class in colourpaletteextractor.view.otherviews), 15

`browse_button` (colourpaletteextractor.view.otherviews.PreferencesWidget attribute), 17

## C

`C_STAR_PERCENTILE` (colourpaletteextractor.model.algorithms.nieves2020.Nieves2020 attribute), 4

`c_stars` (colourpaletteextractor.model.algorithms.cielabcube.CielabCube property), 3

`calculate_mean_colour()` (colourpaletteextractor.model.algorithms.cielabcube.CielabCube method), 3

`cancel_batch_button` (colourpaletteextractor.view.otherviews.BatchGenerationProgressWidget attribute), 15

`change_output_directory()` (colourpaletteextractor.model.model.ColourPaletteExtractorModel method), 10

`change_toggle_recoloured_image_pressed()` (colourpaletteextractor.view.tabview.NewTab method), 20

`CielabCube` (class in colourpaletteextractor.model.algorithms.cielabcube), 3

`close_current_tab()` (colourpaletteextractor.view.mainview.MainView method), 14

`close_temporary_directory()` (colourpaletteextractor.model.model.ColourPaletteExtractorModel

- method), 10
  - closeEvent() (colourpaletteextractor.view.mainview.MainView method), 14
  - colour\_palette (colourpaletteextractor.model.imagedata.ImageData property), 9
  - colour\_palette\_dock (colourpaletteextractor.view.mainview.MainView attribute), 13
  - colour\_palette\_relative\_frequency (colourpaletteextractor.model.imagedata.ImageData property), 9
  - ColourBox (class in colourpaletteextractor.view.tabview), 18
  - ColourPaletteDock (class in colourpaletteextractor.view.tabview), 18
  - colourpaletteextractor
    - module, 21
  - colourpaletteextractor.controller
    - module, 2
  - colourpaletteextractor.controller.controller
    - module, 1
  - colourpaletteextractor.controller.worker
    - module, 1
  - colourpaletteextractor.examples
    - module, 2
  - colourpaletteextractor.examples.generatecolourpaletteexample
    - module, 2
  - colourpaletteextractor.model
    - module, 12
  - colourpaletteextractor.model.algorithms
    - module, 7
  - colourpaletteextractor.model.algorithms.cielabcube
    - module, 3
  - colourpaletteextractor.model.algorithms.nieves2020
    - module, 4
  - colourpaletteextractor.model.algorithms.palettealgorithm
    - module, 6
  - colourpaletteextractor.model.generatereport
    - module, 7
  - colourpaletteextractor.model.imagedata
    - module, 8
  - colourpaletteextractor.model.model
    - module, 9
  - colourpaletteextractor.tests
    - module, 13
  - colourpaletteextractor.tests.helpers
    - module, 12
  - colourpaletteextractor.tests.helpers.helperfunctions
    - module, 12
  - colourpaletteextractor.tests.nieves2020\_test
    - module, 12
  - colourpaletteextractor.view
    - module, 21
  - colourpaletteextractor.view.mainview
    - module, 13
  - colourpaletteextractor.view.otherviews
    - module, 15
  - colourpaletteextractor.view.tabview
    - module, 18
  - ColourPaletteExtractorController (class in colourpaletteextractor.controller.controller), 1
  - ColourPaletteExtractorModel (class in colourpaletteextractor.model.model), 9
  - ColourPaletteReport (class in colourpaletteextractor.model.generatereport), 7
  - continue\_thread (colourpaletteextractor.model.algorithms.palettealgorithm.PaletteAlgorithm property), 6
  - continue\_thread (colourpaletteextractor.model.imagedata.ImageData property), 9
  - convert\_lab\_2\_rgb() (in module colourpaletteextractor.model.algorithms.nieves2020), 5
  - convert\_rgb\_2\_lab() (in module colourpaletteextractor.model.algorithms.nieves2020), 5
  - coordinates (colourpaletteextractor.model.algorithms.cielabcube.CielabCube property), 3
  - create\_new\_tab() (colourpaletteextractor.view.mainview.MainView method), 14
  - create\_report() (colourpaletteextractor.model.generatereport.ReportGenerator method), 8
  - CUBE\_SIZE (colourpaletteextractor.model.algorithms.nieves2020.Nieves2020 attribute), 4
  - current\_tab\_changed() (colourpaletteextractor.controller.controller.ColourPaletteExtractorController method), 1
- ## D
- DEFAULT\_ALGORITHM (colourpaletteextractor.model.model.ColourPaletteExtractorModel attribute), 10
  - DEFAULT\_HEIGHT (colourpaletteextractor.model.model.ColourPaletteExtractorModel attribute), 10
  - default\_new\_tab\_image (colourpaletteextractor.view.mainview.MainView attribute), 15
  - default\_path\_button (colourpaletteextractor.view.otherviews.PreferencesWidget attribute), 17
  - DEFAULT\_USE\_USER\_DIRECTORY (colourpaletteextractor.model.model.ColourPaletteExtractorModel attribute), 10
  - DEFAULT\_USER\_DIRECTORY (colourpaletteextractor.model.model.ColourPaletteExtractorModel attribute), 10
  - DEFAULT\_WIDTH (colourpaletteextractor.model.model.ColourPaletteExtractorModel attribute), 10

## E

elided\_text() (colourpaletteextractor.view.otherviews.ElidedLabel method), 16  
 ElidedLabel (class in colourpaletteextractor.view.otherviews), 16  
 enterEvent() (colourpaletteextractor.view.tabview.ColourBox method), 18  
 error (colourpaletteextractor.controller.worker.WorkerSignals attribute), 2  
 ErrorBox (class in colourpaletteextractor.view.otherviews), 16  
 event() (colourpaletteextractor.view.tabview.ImageDisplay method), 19  
 extension (colourpaletteextractor.model.image-data.ImageData property), 9

## F

file\_name\_and\_path (colourpaletteextractor.model.imagedata.ImageData property), 9  
 finished (colourpaletteextractor.controller.worker.WorkerSignals attribute), 2  
 footer() (colourpaletteextractor.model.generator.report.ColourPaletteReport method), 7

## G

generate\_all\_palette\_action (colourpaletteextractor.view.mainview.MainView attribute), 13  
 generate\_all\_report\_action (colourpaletteextractor.view.mainview.MainView attribute), 13  
 generate\_colour\_palette() (colourpaletteextractor.model.algorithms.nieves2020.Nieves2020 method), 5  
 generate\_colour\_palette() (colourpaletteextractor.model.algorithms.palettealgorithm.PaletteAlgorithm method), 6  
 generate\_colour\_palette\_from\_image() (in module colourpaletteextractor.model.model), 11  
 generate\_palette() (colourpaletteextractor.model.model.ColourPaletteExtractorModel method), 10  
 generate\_palette\_action (colourpaletteextractor.view.mainview.MainView attribute), 13  
 generate\_palette\_available (colourpaletteextractor.view.tabview.NewTab property), 20  
 generate\_report() (in module colourpaletteextractor.model.generator.report), 8  
 generate\_report\_action (colourpaletteextractor.view.mainview.MainView attribute), 13  
 generate\_report\_available (colourpaletteextractor.view.tabview.NewTab property), 20

tor.view.tabview.NewTab property), 20  
 get\_algorithms\_and\_buttons() (colourpaletteextractor.view.otherviews.PreferencesWidget method), 17  
 get\_c\_star\_percentile\_value() (colourpaletteextractor.model.algorithms.cielabcube.CielabCube method), 3  
 get\_c\_stars() (in module colourpaletteextractor.model.algorithms.nieves2020), 6  
 get\_image() (in module colourpaletteextractor.tests.helpers.helperfunctions), 12  
 get\_image\_as\_q\_image() (colourpaletteextractor.model.imagedata.ImageData static method), 9  
 get\_image\_data() (colourpaletteextractor.model.model.ColourPaletteExtractorModel method), 11  
 get\_implemented\_algorithms() (in module colourpaletteextractor.model.algorithms.palettealgorithm), 7  
 get\_l\_star\_percentile\_value() (colourpaletteextractor.model.algorithms.cielabcube.CielabCube method), 3  
 get\_relative\_frequencies() (in module colourpaletteextractor.model.algorithms.cielabcube), 4  
 get\_settings() (in module colourpaletteextractor.model.model), 12  
 get\_slider\_positions() (colourpaletteextractor.view.tabview.NewTab method), 20

## H

header (colourpaletteextractor.view.otherviews.ErrorBox attribute), 16  
 header() (colourpaletteextractor.model.generator.report.ColourPaletteReport method), 7

## I

image (colourpaletteextractor.model.imagedata.ImageData property), 9  
 image\_data\_id\_dictionary (colourpaletteextractor.model.model.ColourPaletteExtractorModel property), 11  
 image\_display (colourpaletteextractor.view.tabview.NewTab attribute), 20  
 image\_id (colourpaletteextractor.view.tabview.NewTab property), 20  
 IMAGE\_START\_POSITION (colourpaletteextractor.model.generator.report.ColourPaletteReport attribute), 7  
 IMAGE\_WIDTH (colourpaletteextractor.model.generator.report.ColourPaletteReport attribute), 7  
 image\_zoom() (colourpaletteextractor.view.tabview.ImageDisplay method), 19  
 ImageData (class in colourpaletteextractor.view.tabview.NewTab property), 20

model.imagedata), 8  
 ImageDisplay (class in colourpaletteextrac-  
 tor.view.tabview), 19  
 increment\_pixel\_count\_after\_reassignment()  
 (colourpaletteextractor.model.algorithms.cielabcube.CielabCube  
 method), 4

## L

l\_stars (colourpaletteextractor.model.algorith-  
 ms.cielabcube.CielabCube property), 4  
 label (colourpaletteextractor.view.otherviews.-  
 BatchGenerationProgressWidget  
 attribute), 15  
 leaveEvent() (colourpaletteextractor.view.tab-  
 view.ColourBox method), 18

## M

MainView (class in colourpaletteextrac-  
 tor.view.mainview), 13  
 MARGIN (colourpaletteextractor.model.gener-  
 atereport.ColourPaletteReport  
 attribute), 7  
 MAX\_IMAGE\_HEIGHT (colourpaletteextrac-  
 tor.model.generatereport.ColourPaletteReport  
 attribute), 7  
 mean\_colour (colourpaletteextractor.model.al-  
 gorithms.cielabcube.CielabCube prop-  
 erty), 4  
 MIN\_L\_STAR (colourpaletteextractor.model.al-  
 gorithms.nieves2020.Nieves2020  
 attribute), 4

## module

colourpaletteextractor, 21  
 colourpaletteextractor.controller, 2  
 colourpaletteextractor.controller.controller,  
 1  
 colourpaletteextractor.controller.worker, 1  
 colourpaletteextractor.examples, 2  
 colourpaletteextractor.examples.generatecolourpaletteexample,  
 2  
 colourpaletteextractor.model, 12  
 colourpaletteextractor.model.algorithms, 7  
 colourpaletteextractor.model.algorithms.cielabcube,  
 3  
 colourpaletteextractor.model.algorithms.nieves2020,  
 4  
 colourpaletteextractor.model.algorithms.palettealgorithm,  
 6  
 colourpaletteextractor.model.generatereport,  
 7  
 colourpaletteextractor.model.imagedata, 8  
 colourpaletteextractor.model.model, 9  
 colourpaletteextractor.tests, 13  
 colourpaletteextractor.tests.helpers, 12  
 colourpaletteextractor.tests.helpers.helperfunctions,  
 12  
 colourpaletteextractor.tests.nieves2020\_test,

12  
 colourpaletteextractor.view, 21  
 colourpaletteextractor.view.mainview, 13  
 colourpaletteextractor.view.otherviews, 15  
 colourpaletteextractor.view.tabview, 18

## N

NAME (colourpaletteextractor.model.algo-  
 rithms.nieves2020.Nieves2020CentredCubes  
 attribute), 5  
 NAME (colourpaletteextractor.model.algo-  
 rithms.nieves2020.Nieves2020OffsetCubes  
 attribute), 5  
 name (colourpaletteextractor.model.algorith-  
 ms.palettealgorithm.PaletteAlgorithm  
 property), 6  
 name (colourpaletteextractor.model.imageda-  
 ta.ImageData property), 9  
 NewTab (class in colourpaletteextrac-  
 tor.view.tabview), 20  
 Nieves2020 (class in colourpaletteextractor.-  
 model.algorithms.nieves2020), 4  
 Nieves2020CentredCubes (class in colourpalet-  
 teextractor.model.algorithms.nieves2020),  
 5  
 Nieves2020OffsetCubes (class in colourpalette-  
 extractor.model.algorithms.nieves2020),  
 5

## O

open\_action (colourpaletteextractor.view.main-  
 view.MainView attribute), 13  
 output\_tab (colourpaletteextractor.view.oth-  
 erviews.PreferencesWidget attribute),  
 17

## P

paintEvent() (colourpaletteextractor.view.oth-  
 erviews.ElidedLabel method), 16  
 PaletteAlgorithm (class in colourpaletteextrac-  
 tor.model.algorithms.palettealgorithm),  
 6  
 pixel\_count\_after\_reassignment (colourpalette-  
 extractor.model.algorithms.cielabcube.CielabCube  
 property), 4  
 pixels (colourpaletteextractor.model.algorith-  
 ms.cielabcube.CielabCube property), 4  
 preferences\_menu\_action (colourpaletteextrac-  
 tor.view.mainview.MainView  
 attribute), 13  
 PreferencesWidget (class in colourpaletteex-  
 tractor.view.otherviews), 16  
 progress (colourpaletteextractor.controller.-  
 worker.WorkerSignals attribute), 2  
 progress\_bar\_value (colourpaletteextrac-  
 tor.view.tabview.NewTab property), 20



## R

recoloured\_image (colourpaletteextractor.model.imagedata.ImageData property), 9  
 relevant (colourpaletteextractor.model.algorithms.cielabcube.CielabCube property), 4  
 remove\_colour\_palette() (colourpaletteextractor.view.tabview.ColourPaletteDock method), 19  
 remove\_image\_data() (colourpaletteextractor.model.model.ColourPaletteExtractorModel method), 11  
 ReportGenerator (class in colourpaletteextractor.model.generatereport), 8  
 RESOURCES\_DIR (colourpaletteextractor.view.mainview.MainView attribute), 14  
 resources\_path (colourpaletteextractor.view.mainview.MainView attribute), 15  
 result (colourpaletteextractor.controller.worker.WorkerSignals attribute), 2  
 run() (colourpaletteextractor.controller.worker.Worker method), 2

## S

save\_report() (colourpaletteextractor.model.generatereport.ReportGenerator method), 8  
 SECONDARY\_THRESHOLD (colourpaletteextractor.model.algorithms.nieves2020.Nieves2020 attribute), 4  
 set\_algorithm() (colourpaletteextractor.model.model.ColourPaletteExtractorModel method), 11  
 set\_cancel\_text() (colourpaletteextractor.view.otherviews.BatchGenerationProgressWidget method), 15  
 set\_progress\_callback() (colourpaletteextractor.model.algorithms.palettealgorithm.PaletteAlgorithm method), 6  
 set\_slider\_positions() (colourpaletteextractor.view.tabview.NewTab method), 20  
 set\_status\_bar() (colourpaletteextractor.view.otherviews.StatusBar method), 18  
 show\_file\_dialog\_box() (colourpaletteextractor.view.mainview.MainView method), 15  
 show\_help\_action (colourpaletteextractor.view.mainview.MainView attribute), 13  
 show\_output\_directory\_dialog\_box() (colourpaletteextractor.view.otherviews.PreferencesWidget method), 17  
 show\_palette\_dock\_action (colourpaletteextractor.view.mainview.MainView

attribute), 14  
 show\_preferences() (colourpaletteextractor.view.otherviews.PreferencesWidget method), 17  
 show\_toolbar\_action (colourpaletteextractor.view.mainview.MainView attribute), 14  
 show\_widget() (colourpaletteextractor.view.otherviews.BatchGenerationProgressWidget method), 15  
 sort\_colour\_palette() (colourpaletteextractor.model.imagedata.ImageData method), 9  
 staticMetaObject (colourpaletteextractor.controller.worker.WorkerSignals attribute), 2  
 staticMetaObject (colourpaletteextractor.view.mainview.MainView attribute), 15  
 staticMetaObject (colourpaletteextractor.view.otherviews.AboutBox attribute), 15  
 staticMetaObject (colourpaletteextractor.view.otherviews.BatchGenerationProgressWidget attribute), 16  
 staticMetaObject (colourpaletteextractor.view.otherviews.ElidedLabel attribute), 16  
 staticMetaObject (colourpaletteextractor.view.otherviews.ErrorBox attribute), 16  
 staticMetaObject (colourpaletteextractor.view.otherviews.PreferencesWidget attribute), 17  
 staticMetaObject (colourpaletteextractor.view.otherviews.StatusBar attribute), 18  
 staticMetaObject (colourpaletteextractor.view.tabview.ColourBox attribute), 18  
 staticMetaObject (colourpaletteextractor.view.tabview.ColourPaletteDock attribute), 19  
 staticMetaObject (colourpaletteextractor.view.tabview.ImageDisplay attribute), 19  
 staticMetaObject (colourpaletteextractor.view.tabview.NewTab attribute), 20  
 status (colourpaletteextractor.view.mainview.MainView attribute), 14  
 status\_bar\_state (colourpaletteextractor.view.tabview.NewTab property), 21  
 StatusBar (class in colourpaletteextractor.view.otherviews), 17  
 stop\_action (colourpaletteextractor.view.mainview.MainView attribute), 13  
 SUPPORTED\_IMAGE\_TYPES (colourpaletteextractor.model.model.ColourPaletteExtractorModel attribute), 10

## T

tabs (colourpaletteextractor.view.mainview.-  
MainView attribute), 13

test\_closest\_relevant\_colour\_used\_to\_recolour\_pixel()  
(in module colourpaletteextractor.test-  
s.nieves2020\_test), 12

test\_cube\_colour\_must\_occur\_more\_than\_three\_percent  
(in module colourpaletteextractor.test-  
s.nieves2020\_test), 12

test\_cube\_colour\_must\_occur\_more\_than\_three\_percent  
(in module colourpaletteextractor.test-  
s.nieves2020\_test), 12

test\_low\_a\_b\_colour\_does\_not\_meet\_secondary\_requirements  
(in module colourpaletteextractor.test-  
s.nieves2020\_test), 12

test\_low\_a\_b\_colour\_does\_not\_meet\_secondary\_requirements  
(in module colourpaletteextractor.test-  
s.nieves2020\_test), 12

test\_nieves2020\_centred\_cubes\_constructor()  
(in module colourpaletteextractor.test-  
s.nieves2020\_test), 12

test\_nieves2020\_offset\_cubes\_constructor() (in  
module colourpaletteextractor.test-  
s.nieves2020\_test), 12

test\_primary\_requirements\_1() (in module  
colourpaletteextractor.tests.nieves2020\_test),  
12

test\_primary\_requirements\_2() (in module  
colourpaletteextractor.tests.nieves2020\_test),  
12

test\_primary\_requirements\_3() (in module  
colourpaletteextractor.tests.nieves2020\_test),  
12

test\_recoloured\_image\_of\_same\_size\_1() (in  
module colourpaletteextractor.test-  
s.nieves2020\_test), 12

test\_recoloured\_image\_two\_colours\_1() (in  
module colourpaletteextractor.test-  
s.nieves2020\_test), 12

test\_recoloured\_image\_two\_colours\_2() (in  
module colourpaletteextractor.test-  
s.nieves2020\_test), 12

test\_two\_colours\_in\_same\_cube\_can\_meet\_secondary  
(in module colourpaletteextractor.test-  
s.nieves2020\_test), 13

THRESHOLD (colourpaletteextractor.model.al-  
gorithms.nieves2020.Nieves2020  
attribute), 4

toggle\_recoloured\_image\_action (colourpalet-  
teextractor.view.mainview.MainView  
attribute), 14

toggle\_recoloured\_image\_available (colour-  
paletteextractor.view.tabview.NewTab  
property), 21

toggle\_recoloured\_image\_pressed (colourpalet-  
teextractor.view.tabview.NewTab  
property), 21

tools (colourpaletteextractor.view.mainview.-  
MainView attribute), 14

## U

update\_image() (colourpaletteextrac-  
tor.view.tabview.ImageDisplay  
method), 19

update\_preferences() (colourpaletteextrac-  
tor.view.otherviews.PreferencesWidget  
method), 17

update\_progress\_bar() (colourpaletteextrac-  
tor.view.otherviews.BatchGenerationProgressWidget  
method), 16

update\_status\_bar() (colourpaletteextrac-  
tor.view.otherviews.StatusBar  
method), 18

URL (colourpaletteextractor.model.algorithm-  
s.nieves2020.Nieves2020CentredCubes  
attribute), 5

URL (colourpaletteextractor.model.algorithm-  
s.nieves2020.Nieves2020OffsetCubes  
attribute), 5

url (colourpaletteextractor.model.algorithm-  
palettealgorithm.PaletteAlgorithm  
property), 7

user\_path\_button (colourpaletteextrac-  
tor.view.otherviews.PreferencesWidget  
attribute), 17

user\_path\_selector (colourpaletteextrac-  
tor.view.otherviews.PreferencesWidget  
attribute), 17

## W

wheelEvent() (colourpaletteextractor.view.tab-  
view.NewTab method), 21

Worker (class in colourpaletteextractor.con-  
troller.worker), 1

WorkerSignals (class in colourpaletteextractor.-  
controller.worker), 2

write\_default\_settings() (colourpaletteextrac-  
tor.model.model.ColourPaletteExtractorModel  
method), 11

## Z

zoom\_factor (colourpaletteextractor.view.tab-  
view.ImageDisplay attribute), 19

zoom\_in() (colourpaletteextractor.view.tab-  
view.ImageDisplay method), 19

zoom\_in\_action (colourpaletteextractor.view.-  
mainview.MainView attribute), 14

zoom\_level (colourpaletteextractor.view.tab-  
view.NewTab property), 21

zoom\_out() (colourpaletteextractor.view.tab-  
view.ImageDisplay method), 20

zoom\_out\_action (colourpaletteextractor.view.-  
mainview.MainView attribute), 14

zoom\_out\_factor (colourpaletteextrac-

tor.view.tabview.ImageDisplay  
attribute), [20](#)

