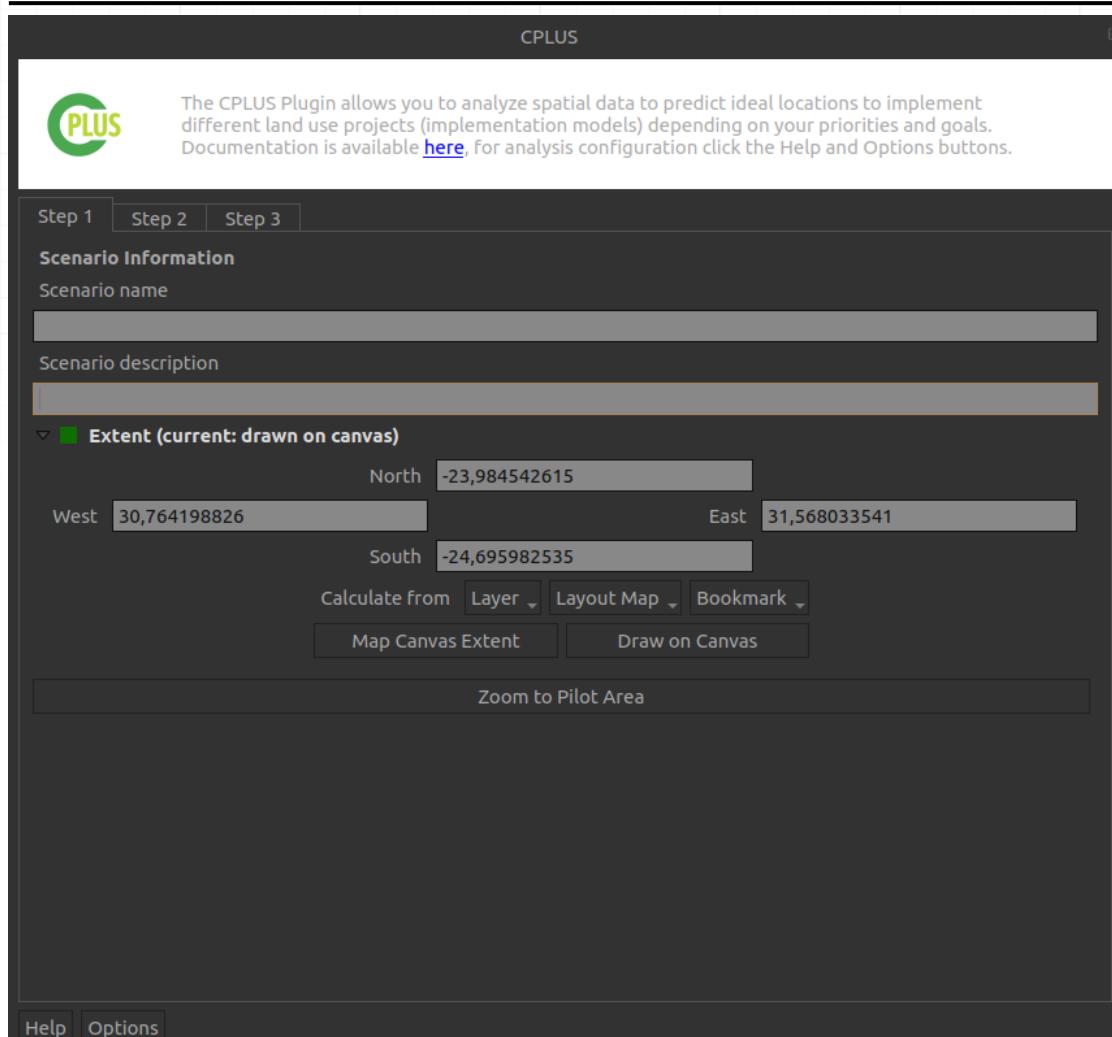




CPLUS QGIS Plugin

Conservation International
2022

1 CPLUS QGIS plugin



1.1 Introduction

The challenge of mitigating climate change and achieving global and national climate targets requires innovative and holistic approaches. In this pursuit, the Climate Positive Land Use Strategy (CPLUS) decision support tool has emerged as a crucial resource. CPLUS is a spatially-explicit roadmap designed to guide land-use planning strategies, utilizing natural climate solutions to drive meaningful and sustainable change. The CPLUS decision support tool combines open-source technology, localized data sets, and modelled products to empower policymakers, land managers, and stakeholders in making informed decisions. By integrating spatial information, such as land cover, carbon stocks, and potential

for carbon sequestration, CPLUS enables the identification of key areas for intervention and investment. By prioritizing these nature-based interventions, CPLUS seeks to harness the power of ecosystems and optimize their climate mitigation potential.

One of the distinguishing features of CPLUS is its ability to address both global and national climate targets. While global climate targets provide a broad framework for action, national targets require context-specific strategies tailored to the unique characteristics of each country. The CPLUS decision support tool considers these diverse factors and assists in designing land-use planning strategies that align with national commitments while contributing to global climate goals. Furthermore, CPLUS recognizes that effective land-use planning involves collaboration and engagement among various stakeholders. It fosters dialogue and cooperation between governments, local communities, indigenous groups, conservation organizations, and private entities, facilitating the development of inclusive and equitable solutions. By involving diverse perspectives and expertise, CPLUS ensures that the decision-making process is participatory and informed by local knowledge.

Piloted in the Bushbuckridge Municipality in the Kruger to Canyons Biosphere of South Africa, the CPLUS framework was tested with a diverse set of stakeholders to identify land use priorities and understand the carbon benefits and biodiversity, ecosystem services co-benefits of different scenarios.

1.2 CPLUS model

1.2.1 Implementation models

Figure 1 shows a flow diagram of the CPLUS analysis model.

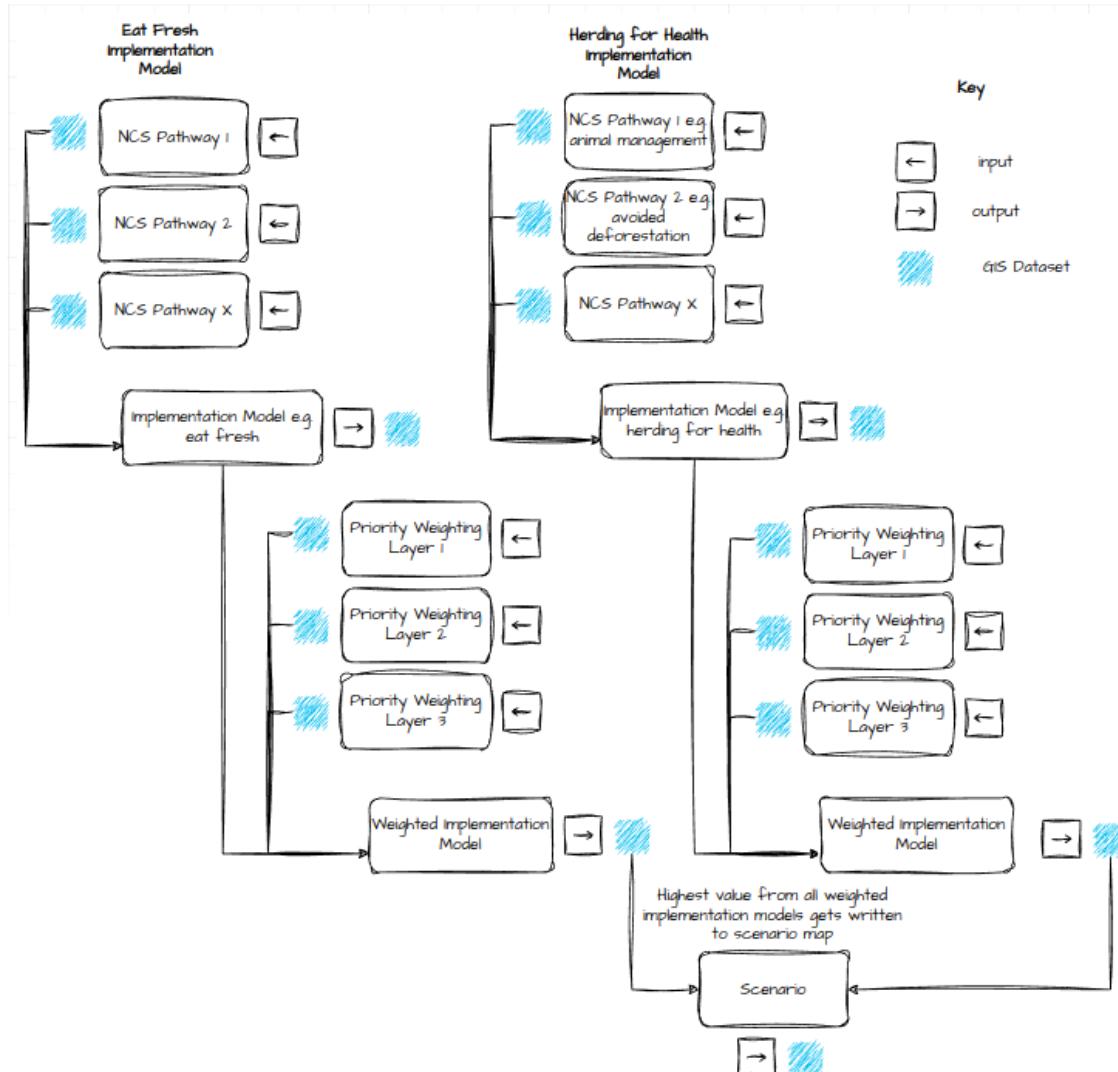


Figure 1: Simplified analysis model

1.2.2 Algorithms

Shown in **Figure 2** is the algorithms applied by the CPLUS model analysis model.

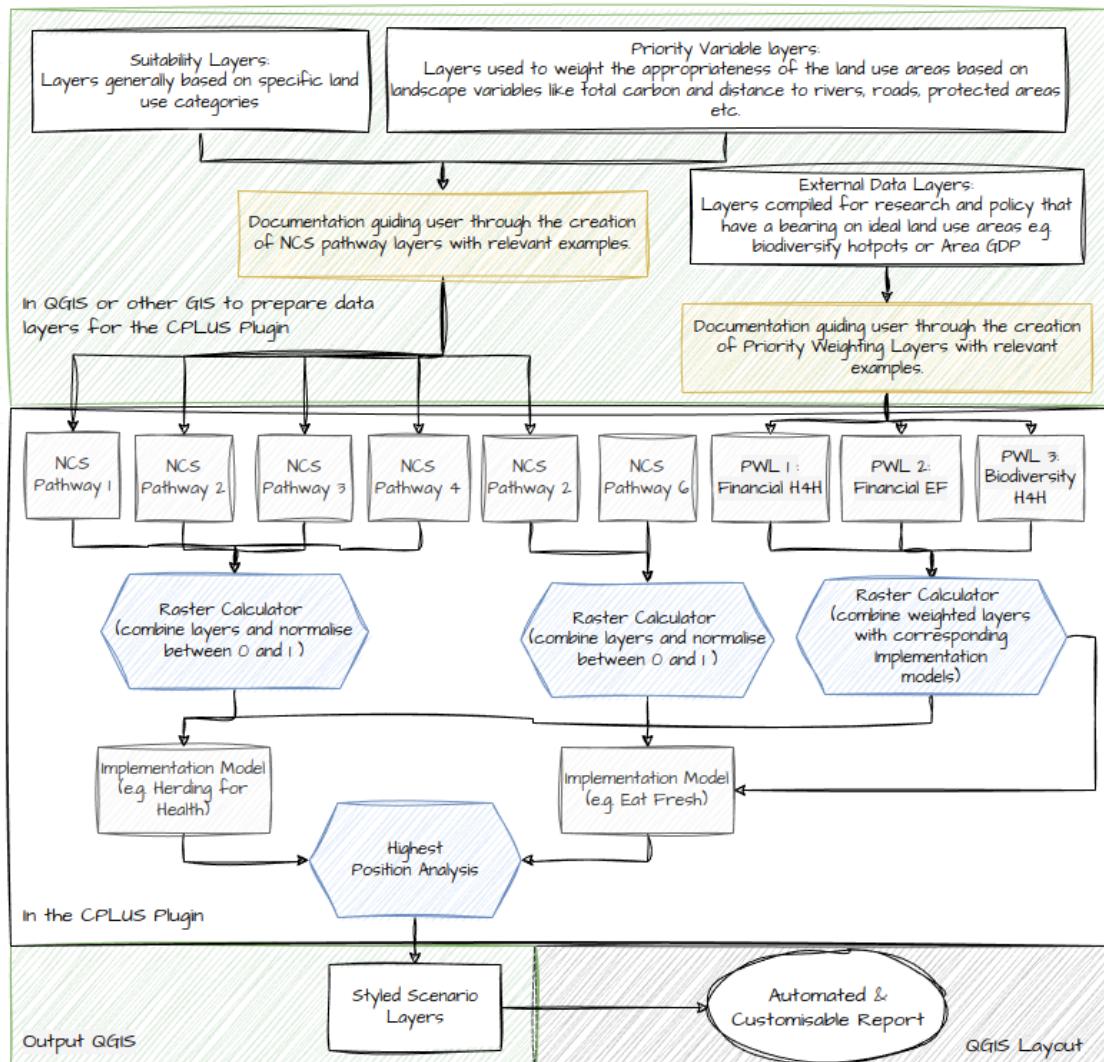


Figure 2: CPLUS simplified analysis workflow with algorithms

1.2.3 References

- <https://www.pnas.org/doi/10.1073/pnas.1710465114>
- <https://royalsocietypublishing.org/doi/10.1098/rstb.2019.0126>

1.3 Site contents

The aim of this site is to provide details on the models and algorithms used by the CPLUS plugin, help a user on using the plugin, guide administrators on testing and providing feedback for the plugin, and provide guidance to developers who want to contribute to the plugin.

Here is a quick introduction on the sections on the site:

- **Users:** Help for users of the plugin
 - [Quick Start](#): Installation guide and how to get started with the plugin
 - [Guide](#): Detailed workflow on how to use the plugin
 - [Manual](#): Descriptions of each UI element
- **Administrators:** Help for an administrator of the plugin
 - [Guide](#): Detailed guide for administrative requirements
 - [Repository](#): Downloadable versions of the plugin
- **Developers:** People who want to contribute towards improving the plugin
 - [Setup](#): How to set up the developers environment for the plugin
 - [Architecture](#):
 - [Documentation](#): Help on adding towards the API documentation for a developer
 - [API](#): Application programming interface for the plugin
- **About:** Information on CI and other contributors to the plugin

1 Users

1.1 Users

The following sections aims at guiding and helping a user on how to use the plugin. This is split into three sections:

- The [quickstart tutorial](#) shows how to install the plugin and help the user to get familiar with the platform.
- The [user guide](#) details common workflows in a tutorial format.
- The [user manual](#) describes the user interface and the various options of the plugin.

1.2 Quick start

Instructions for a user on how to get the plugin working. First section deals with QGIS, and what versions are best to work with. This is followed by a section on how to install the CPLUS plugin, ending with a short tutorial accompanied by a video to show the user how to use the plugin.

1.2.1 QGIS version

The CPLUS plugin might have issues with older versions of QGIS. QGIS 3.32 or higher has thoroughly been tested with no major known issues. Here is a list of possible issues which may occur if using an outdated version of QGIS are being used:

- Processing stalls and does not continue
- Symbolologies (e.g. colour ramps) cannot be applied to the output layers
- Reports cannot be generated

Best will be to update QGIS to the latest version, even for other plugins, as bugs and improvements will be available for those versions. To update QGIS, a user can do the following:

- Go to the [QGIS download webpage](#)
- Download or follow the instructions for the OS on which QGIS needs to be installed
- If a specific version (not the latest version) of QGIS needs to be installed, click on the **All releases** tab

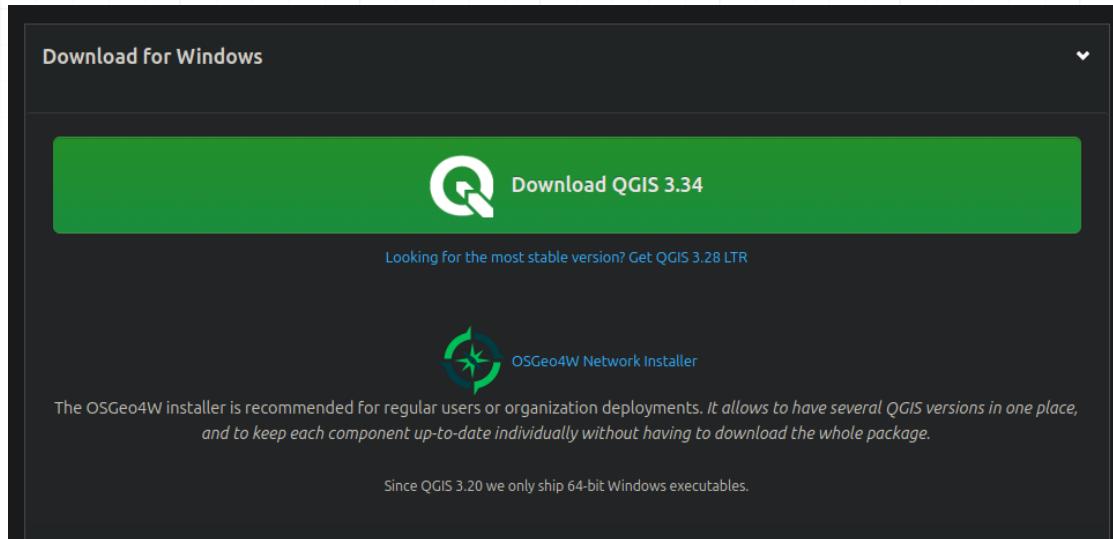
The screenshot shows a dark-themed interface for the OSGeo4W network installer. At the top, there are three tabs: 'INSTALLATION DOWNLOADS' (which is selected), 'ALL RELEASES' (highlighted in blue), and 'SOURCES'. Below these tabs is a list of download options:

- Download for Windows
- Download for macOS
- Download for Linux
- Download for BSD
- Apps for mobile and tablet

Each option is preceded by a small dropdown arrow icon. At the bottom of the list, the text 'OSGeo4W network installer' is visible.

This section will only be helpful for Windows users. The OSGeo4W network installer provides much more option in a user-friendly UI. Advantages of using this installer:

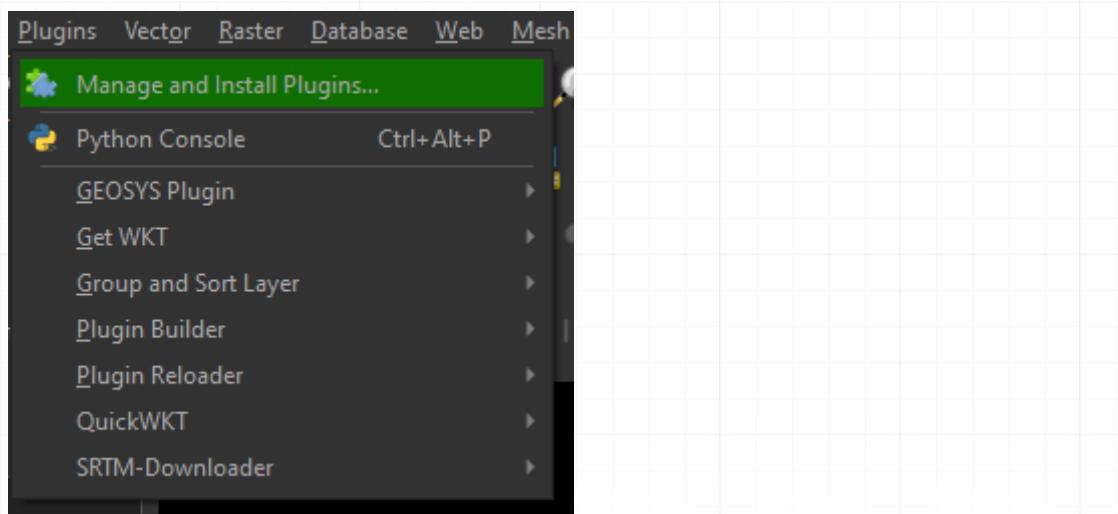
- Easily allows a user to install both the latest and the latest LTR versions of QGIS
- User can select the exact libraries and versions they want to install (e.g. gdal)
- Option to include (or not include) GRASS and SAGA
- Extensions for QGIS
- If a user wants to update an existing QGIS installation, simply run the osgeo.exe in the installation directory
- Simply run the osgeo.exe to update already installed libraries, or install new libraries
- Easy to remove libraries or extensions
- Python modules
- And much more



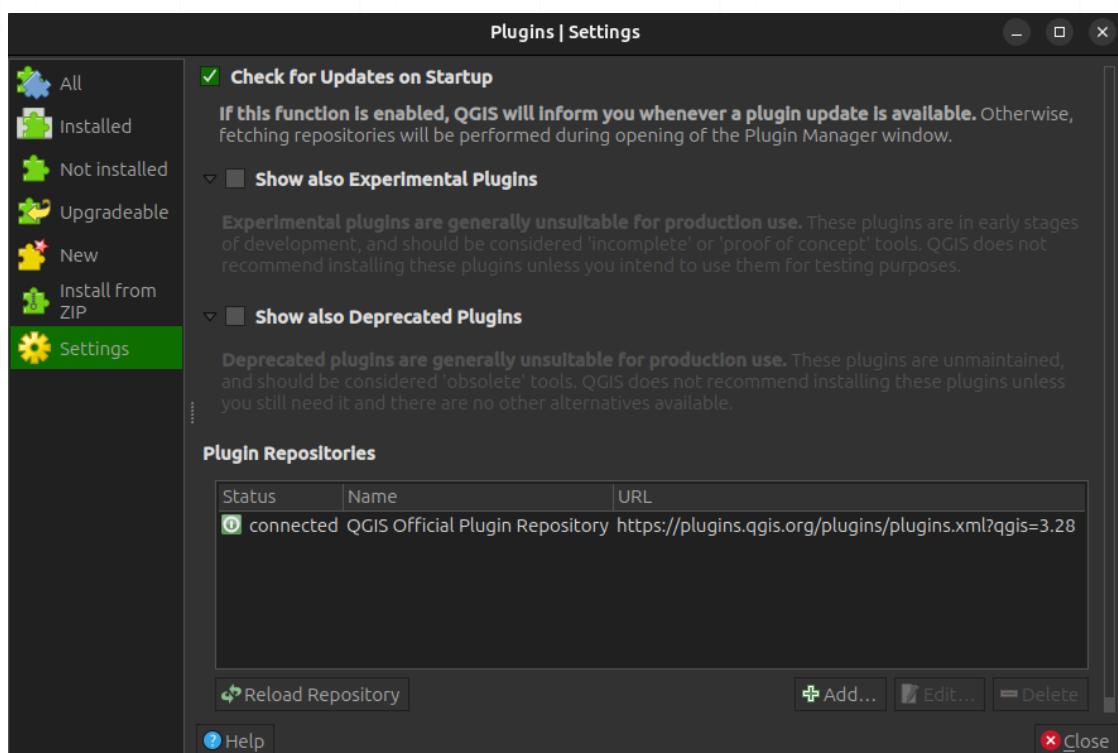
1.2.2 Installation Plugin repository

During the development phase the plugin is available to install via a dedicated plugin repository. This link should be used: <https://raw.githubusercontent.com/kartoza/cplus-plugin/release/docs/repository/plugins.xml>

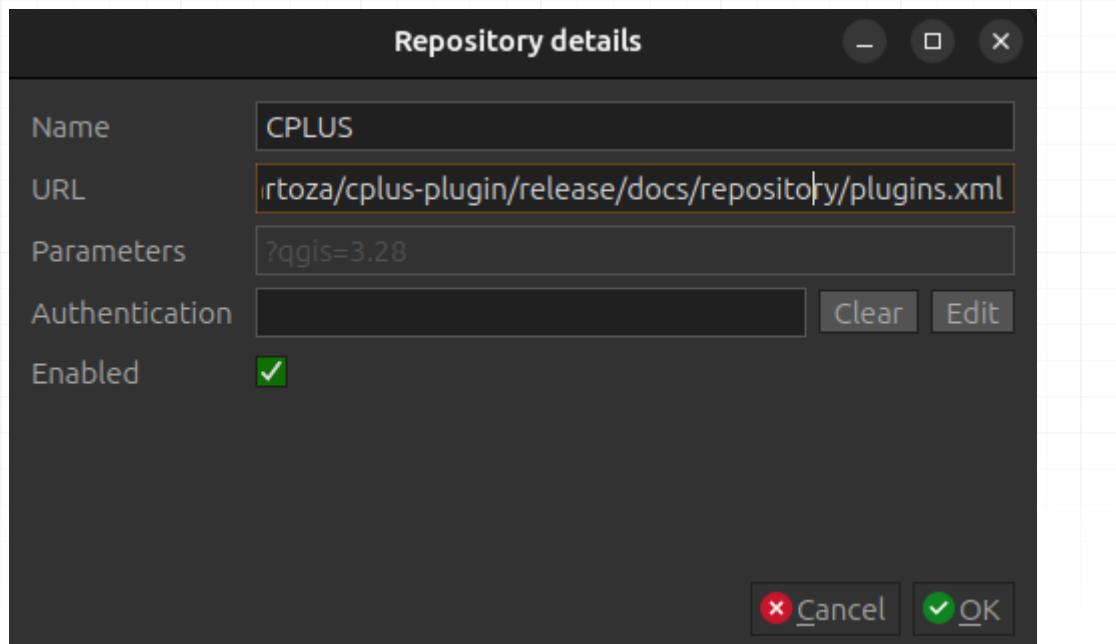
- Open QGIS application and open plugin manager.
- Click on **Plugins -> Manage and Install Plugins** (see **Figure 1**)



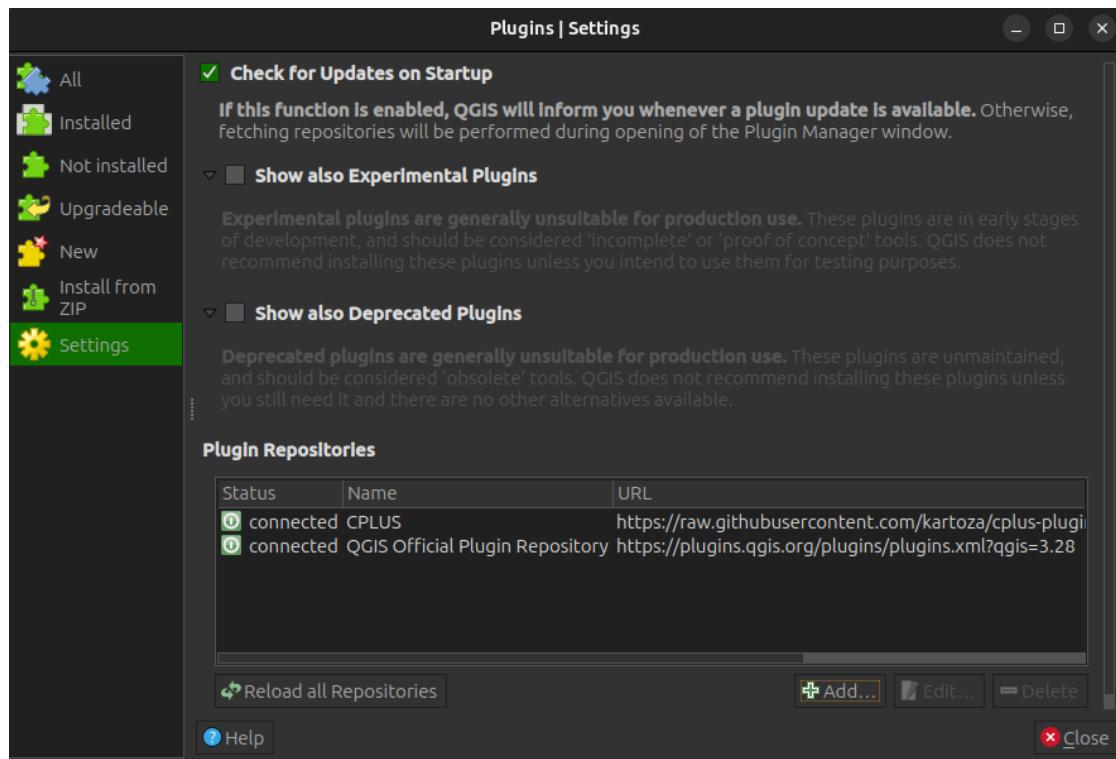
- Open the **Settings** tab



- Click the **Add** button
- Provide the following parameters:
 - **Name:** Provide a name for the repository
 - **URL:** Paste the above repository URL



- Click **OK**
- The result should be similar to the following:



Install from QGIS plugin repository

- Open QGIS application and open plugin manager.
- Click on **Plugins -> Manage and Install Plugins** (see **Figure 1**)

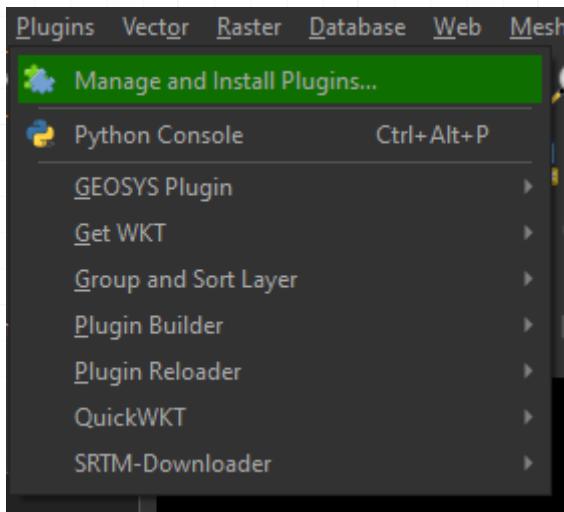


Figure 1: QGIS plugin manager

- Search for **CPLUS** in the **All** page of the plugin manager.
- From the found results, click on the **CPLUS** result item and a page with plugin information will show up.
- Click the **Install Plugin** button at the bottom of the dialog to install the plugin.

Install from ZIP file

Alternatively the plugin can be installed using **Install from ZIP** option on the QGIS plugin manager.

- Download zip file from the required plugin released version <https://github.com/kartoza/cplus-plugin/releases/download/{tagname}/cplus.{version}.zip>
- Open QGIS application and open plugin manager
- Click on **Plugins -> Manage and Install Plugins**
- Click on **Install from ZIP (Figure 2)**
- Select the zip file which contains the plugin
- Click **Install Plugin**

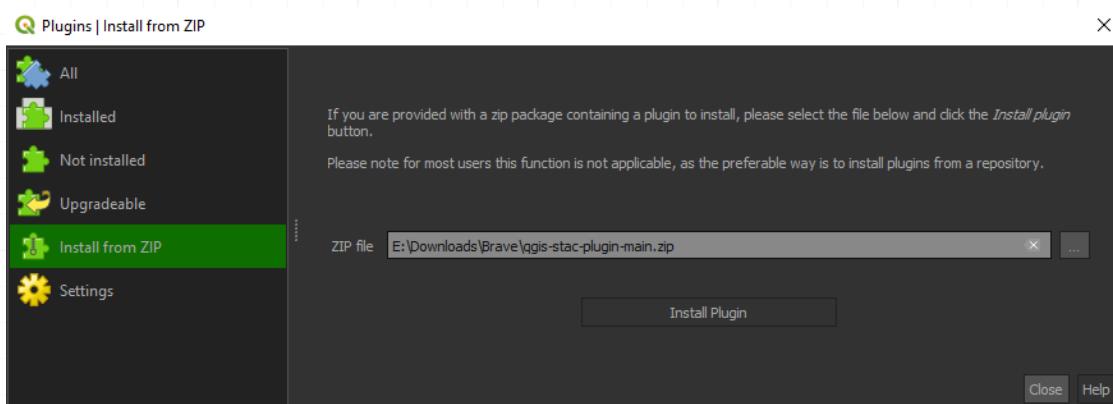


Figure 2: Plugin manager install from zip option

- From the **Install from ZIP** page, select the zip file and click the **Install** button to install plugin

Install from custom plugin repository

Current repository: <https://raw.githubusercontent.com/kartoza/cplus-plugin/release/docs/repository/plugins.xml>

- Open the QGIS plugin manager, then select the **Settings** page (**Figure 3**)

NOTE: The plugin is currently in experimental phase, so enable **Show also Experimental Plugins**

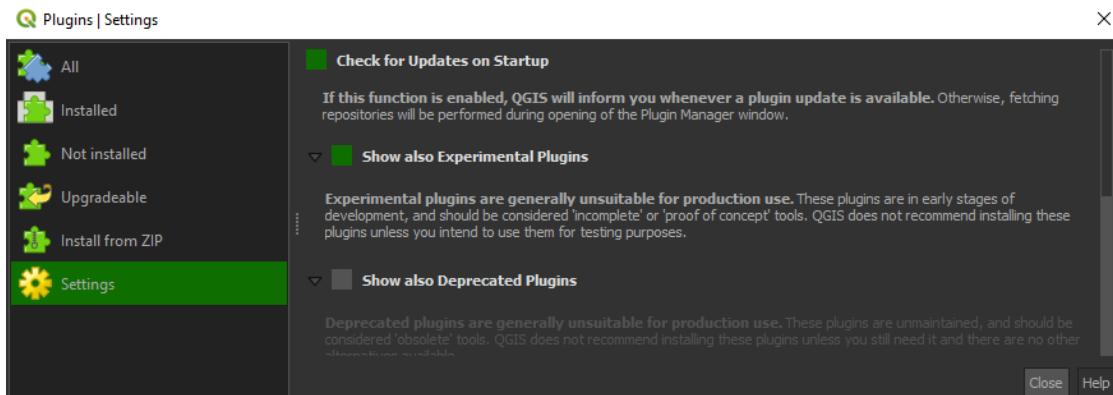


Figure 3: Custom repository installation

- Click **Add** button on the **Plugin Repositories** group box and use the above url to create the new plugin repository.
- The plugin should now be available from the list of all plugins that can be installed.

Disable QGIS official plugin repository in order to not fetch plugins from it.

NOTE: While the development phase is on going the plugin will be flagged as experimental, make sure to enable the QGIS plugin manager in the **Settings** page to show the experimental plugins in order to be able to install it.

When the development work is complete the plugin will be available on the QGIS official plugin repository.

1.2.3 Short tutorial

Short example (**Figure 4**) on how to set parameters in step 1, implementation models in step 2, and weighing in step 3. For a more detailed instructions on how to use the plugin, see the [guide](#) and the [manual](#).

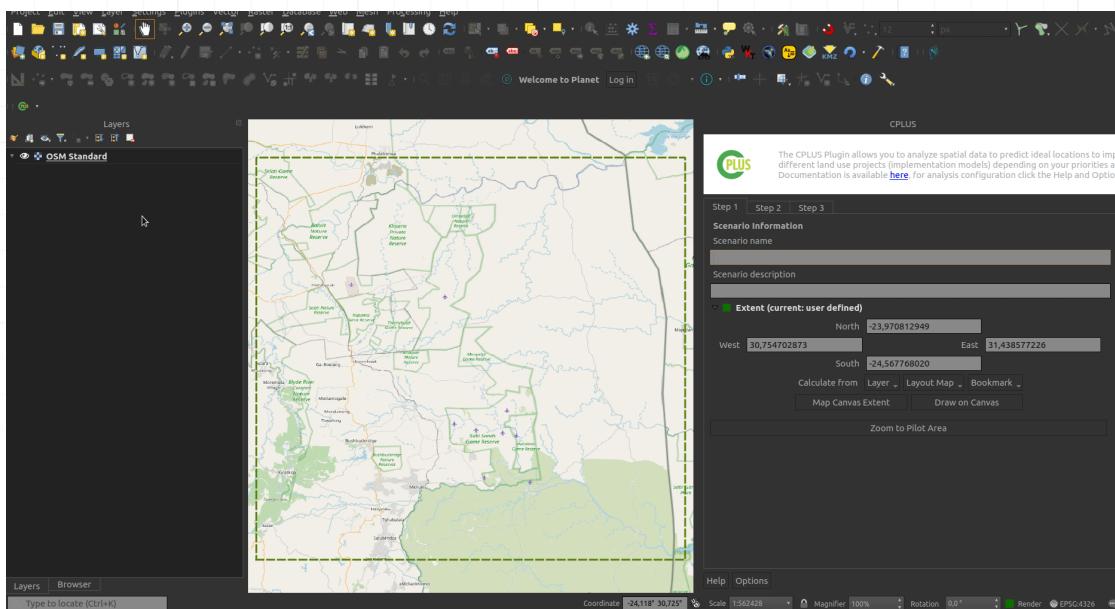


Figure 4: Quick guide on how to use the plugin

1.3 Guide

1.3.1 Preparing data

Data preparation is an important step prior to performing data analysis. This is especially true for spatial data (rasters and vector layers), as there is a lot of factors which plays a role in the end result. Here are some factors which needs to be considered:

- The data should cover the same spatial extent or overlap each other
- Coordinate systems are very important when it comes to the accuracy of your spatial analysis. For most analysis a projected coordinate system (e.g. UTM, Albers Equal Area Conic, South African LO-system, etc) is preferred above a geographic coordinate system (WGS84, Hartebeesthoek84, etc). This is because calculating distances and areas is much more accurate with projected coordinate systems
- Best practice will be to make use of the same coordinate system for each layer. Having a geographic coordinate for some layers, and projected coordinate systems for other, can have negative impacts on your results
- When working with rasters, be sure that the nodata value is set correctly, otherwise the nodata value will be unknown during analysis and will be considered as a normal pixel value
- The plugin can only work with raster layers. If you have data in vector format, consider converting it to a raster
- Any outlier values needs to be removed from the spatial data prior to performing analysis

Taking into account the above can greatly improve the analysis and the results produced from the analysis. This section will further deal with how to prepare your data using tools available in QGIS.

- Click **Processing -> Toolbox** to open the QGIS toolbox
- The toolbox will be used for each section

Carbon and Priority weighted layers

Both Carbon and Priority weighted layers (PWL) should not contain any nodata values. If the nodata pixels is not removed from the rasters, the user's analysis will be less efficient and likely result in a reduction in results (e.g. all nodata pixels will end up as nodata). **Figure 1** shows a Carbon raster with nodata pixels.

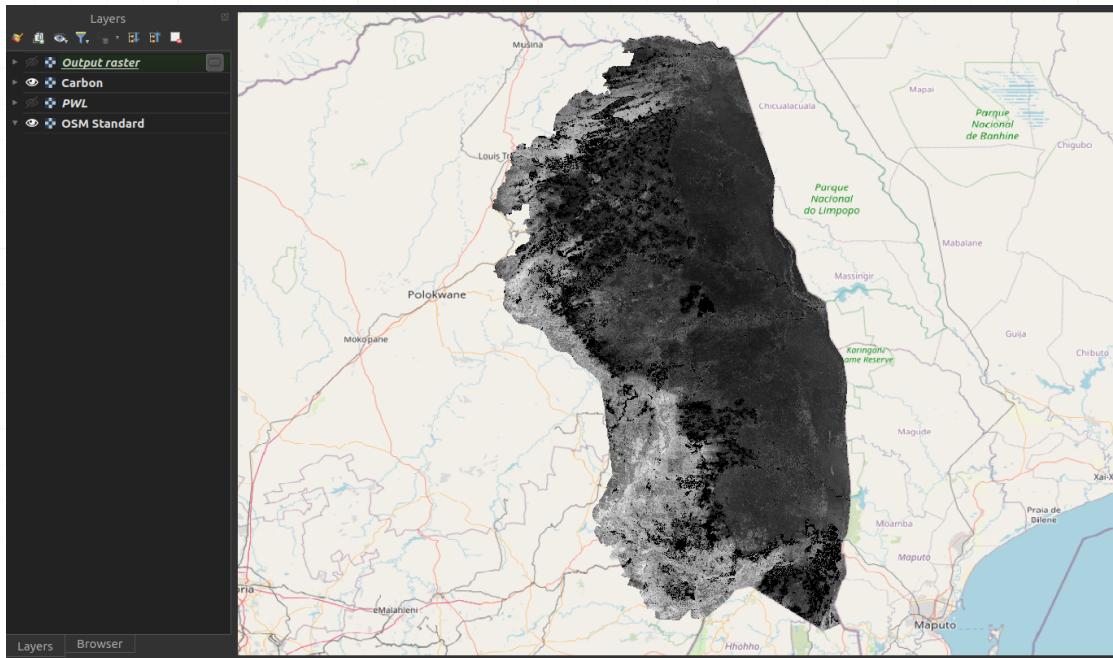


Figure 1: Raster with nodata pixels

Follow these easy steps to remove nodata pixels from a raster:

- In the toolbox search, type "fill nodata"
- Open the tool Fill NoData cells
- Provide the parameters as follows:
 - **Raster input:** Raster layer with nodata pixels which should be removed
 - **Fill value:** Zero should suffice for most cases
 - **Output raster:** Directory to which the filled raster should be stored

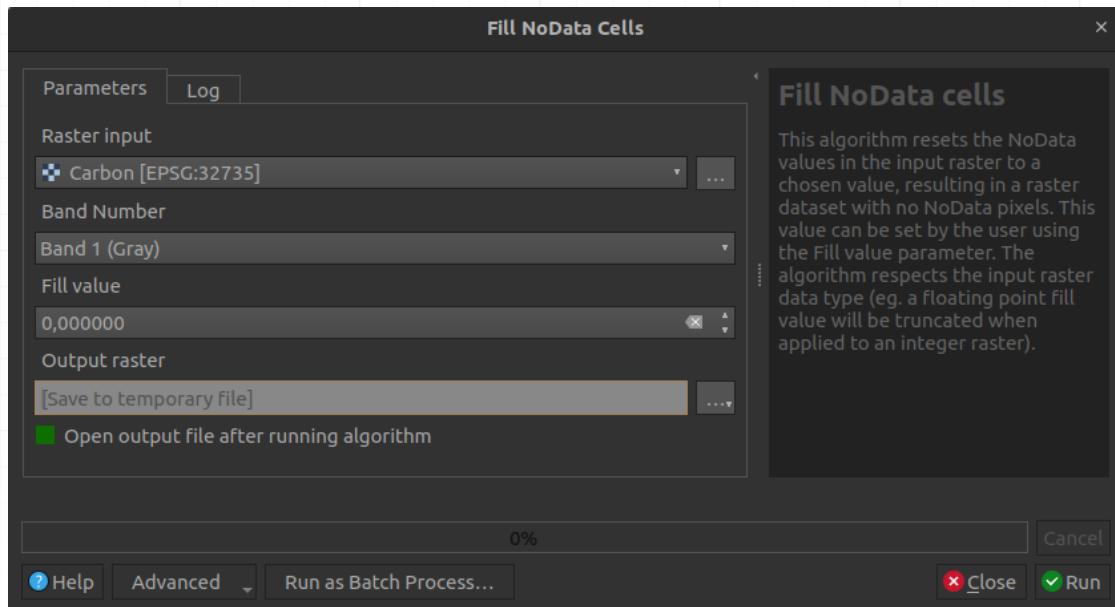


Figure 2: QGIS Fill nodata cells

- Click **Run**

Figure 3 shows a nodata filled raster.

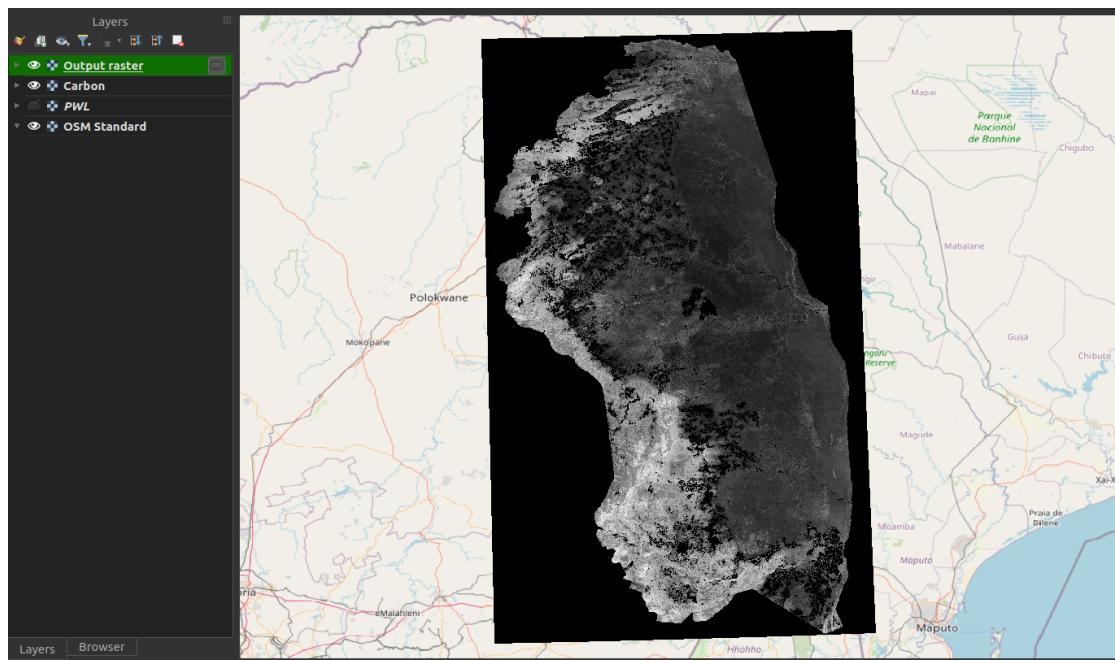


Figure 3: Raster with nodata pixels remove/filled

Coordinate systems

Fix layers with an undefined CRS

Sometimes a spatial dataset might not have its coordinate system defined. This can cause issues and needs to be resolved prior to perform analysis. An unknown coordinate system can be identified as follows:

- Open the layer in QGIS
- QGIS will show a warning next to the layer
- This warning will explain that the coordinate system is not defined

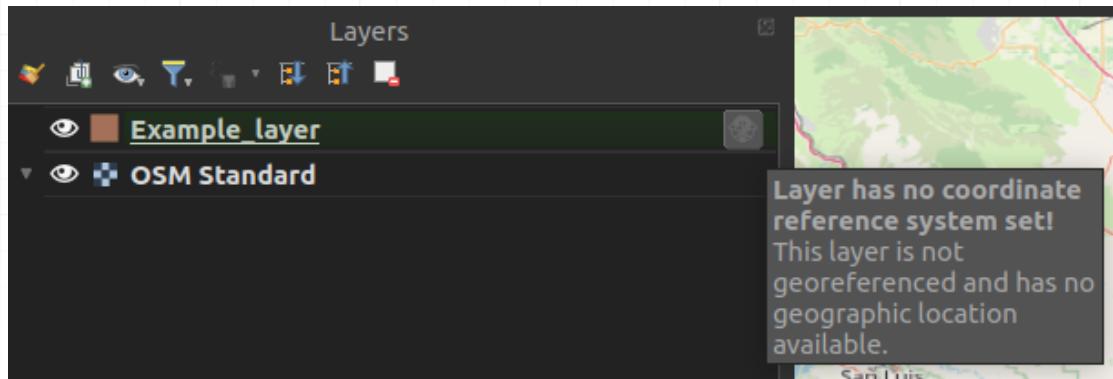


Figure 4: Unknown CRS for a layer in QGIS

- Further investigation can be done by right-clicking on the layer and select **Properties**
- Click on the **Information** tab
- Scroll down to Coordinate Reference System (CRS)
- **Unknown** will be shown if the CRS is not set

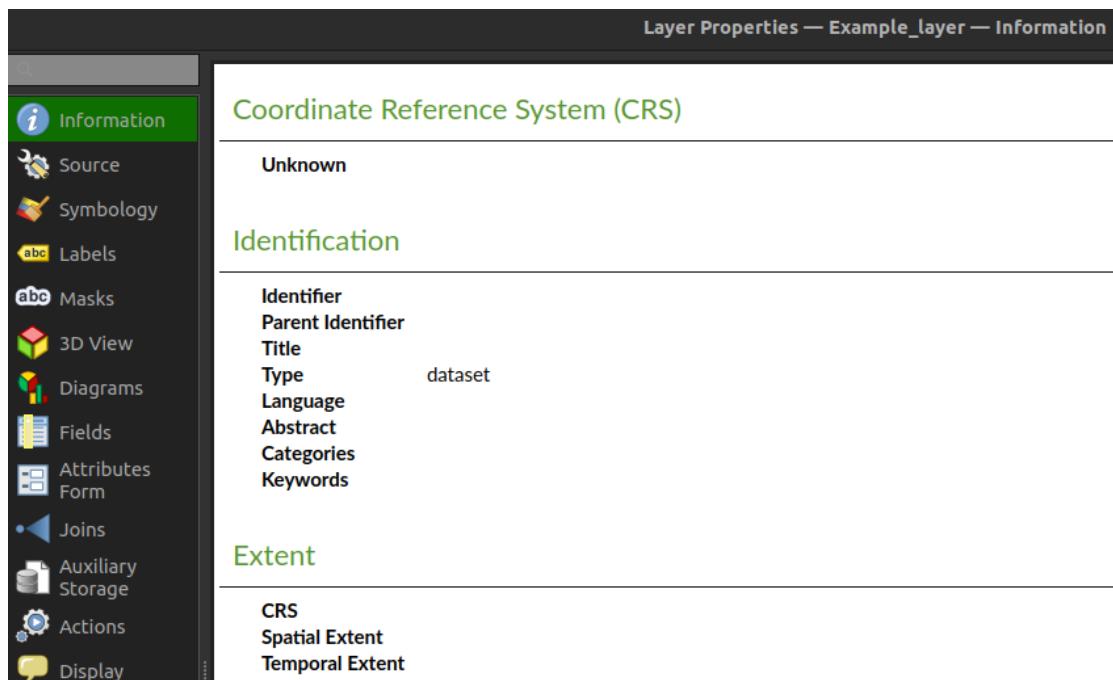


Figure 5: QGIS layer properties

To define the CRS, do the following:

- Type "assign projection" in the toolbox search
- Open the Assign Projection tool in the Raster Projections section
 - If its a vector layer, open the Assign Projection tool in the Vector general section
- Set the parameters as follows:
 - **Input layer:** Layer which has an undefined CRS
 - **Desired CRS:** CRS which the layer coordinates is using
- Click **Run**
- Check if the layer is at its correct position in the QGIS canvas

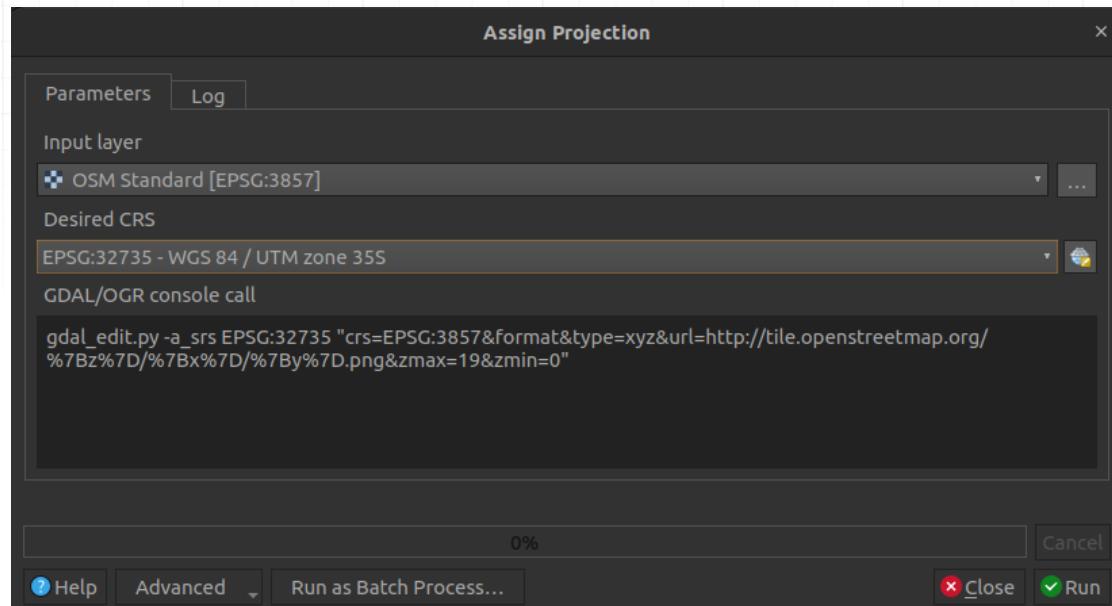


Figure 6: QGIS Assign projection tool

 **Note:**

Be sure you are using the correct coordinate system when defining an unknown coordinate system to a layer. If the incorrect coordinate system is selected, the data will likely not be at the correct position spatially.

Reprojecting (Warping)

- Best will be to convert each dataset in a geographic coordinate system to a projected coordinate system
- Type warp in the QGIS toolbox search
- Under **Raster projections**, select **Warp**

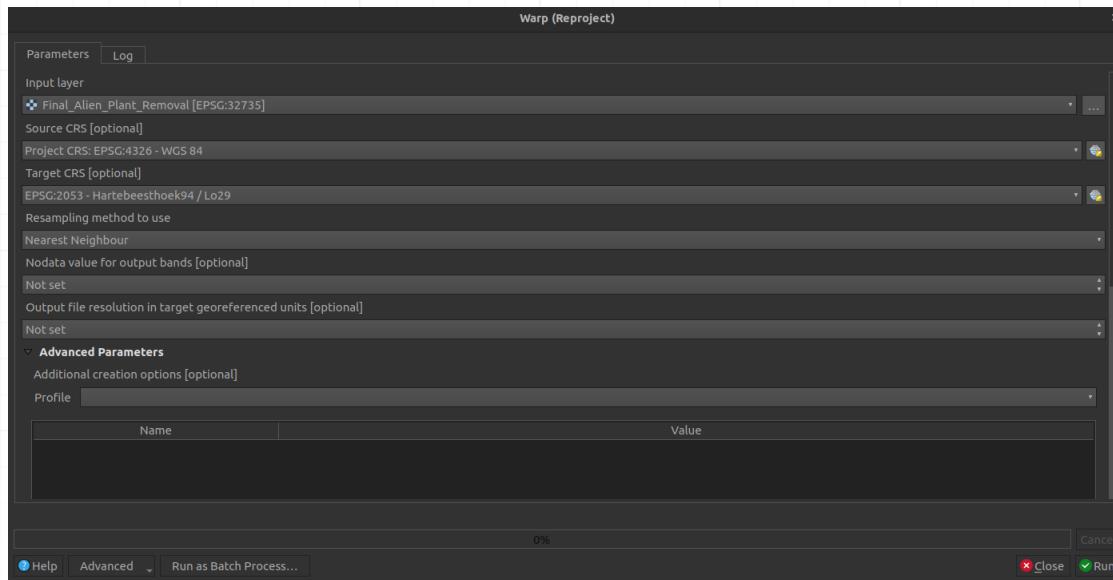


Figure 7: QGIS Warp tool

- Provide the following parameters:
 - **Input layer:** Layer the user wants to reproject
 - **Source CRS:** Current CRS of the layer
 - **Target CRS:** The CRS to what the layer should be projected
 - **Resampling method to use:** Nearest Neighbour. Using other options will change pixel values, which we don't want
 - **Nodata value:** Leave empty, except if the user wants to change the nodata value
 - **Reprojected:** The output file
- Click **Run**
- Do this for all geographic coordinate system rasters
- As mentioned above, best will be for all layers to make use of the same coordinate system

Nodata value

If a nodata value for a raster is not set correctly, it will be considered as a pixel value which is part of the analysis. This can have a negative impact on the analysis results.

How to check if a raster's nodata is set correctly

- Right-click on the raster in QGIS
- Select **Properties**
- Select the **Information** tab
- Scroll down to the **Bands** section
- Under **No-data** there should be a value
- If there is no value, this means that the nodata is not set correctly and therefore needs to be fixed

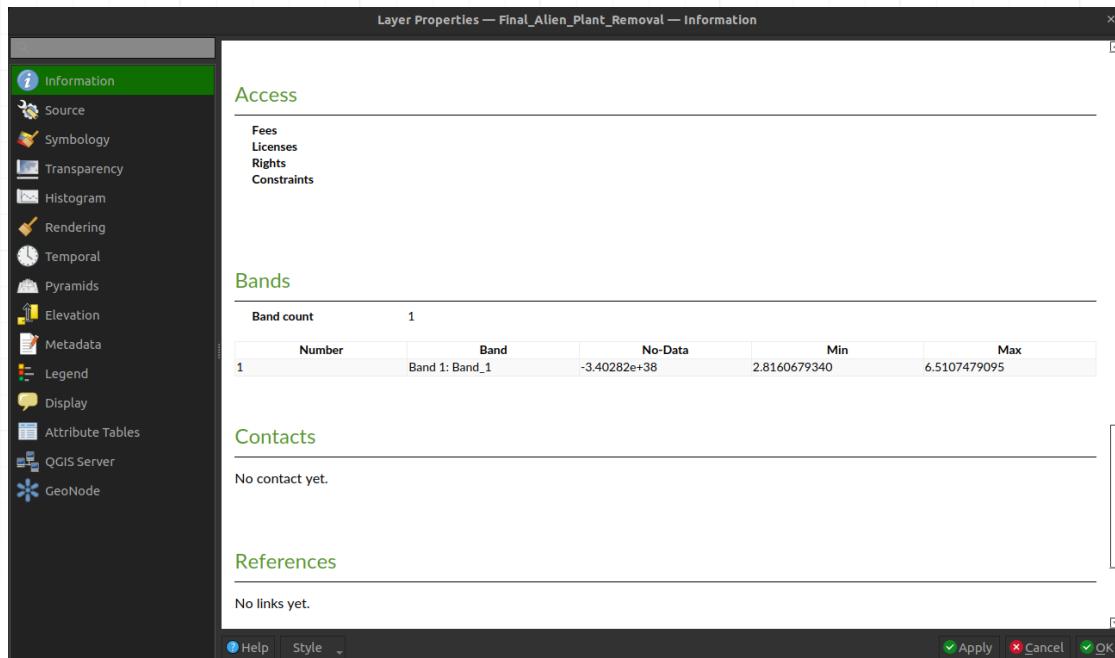


Figure 8: Layer properties to check for nodata value

To fix a nodata issue, do the following:

- Type Translate in the toolbox search
- Open the Translate tool under Raster Conversion

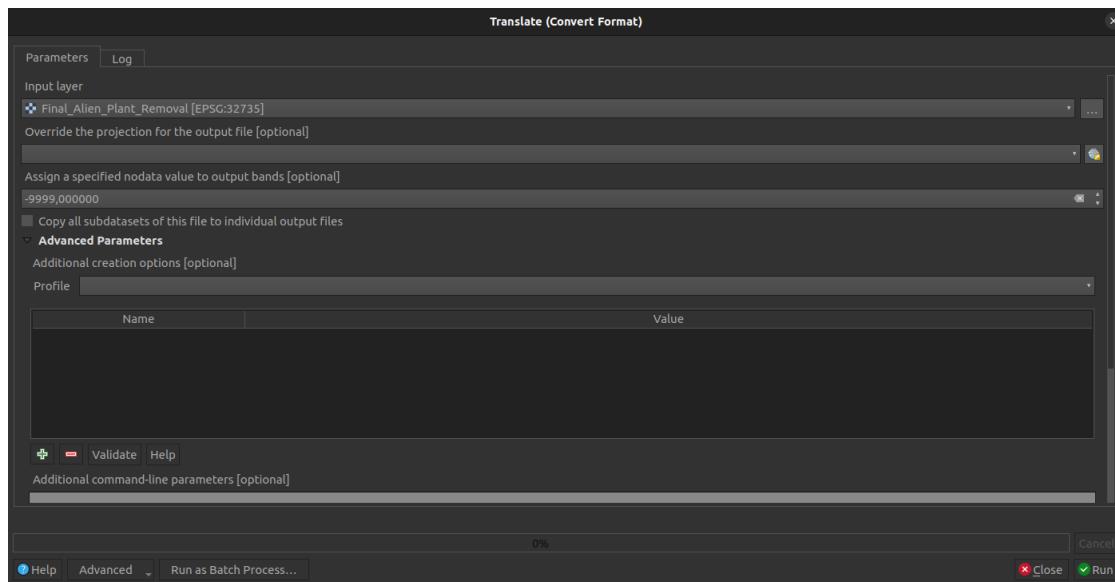


Figure 9: QGIS Translate tool

- Provide the following parameters:
 - **Input layer:** Raster layer
 - **Assign a specific nodata value to output bands:** Provide a desired value here. -9999 will suffice for most cases
 - **Converted:** Output raster

This should solve a nodata issue with a raster. The Translate tool is to convert a raster to another format, but the user can still make use of the same format. This tool is useful to correctly set nodata values when needed.

Outlier pixels/values

A user must check if the raster data only include pixel values within the range it should be. If there are any pixels values outside the range of accepted values, those pixels needs to be removed. This can be accomplished using the Reclassify by table tool.

- Type reclassify by table in the QGIS toolbox search
- Select the Reclassify by table tool
- Set the parameters as follows:
 - **Raster layer:** Layer to be reclassified
 - **Band number:** Like the first band
 - **Reclassified raster:** Output raster
 - **Reclassification table:** Rules for the reclassification (explanation follows)

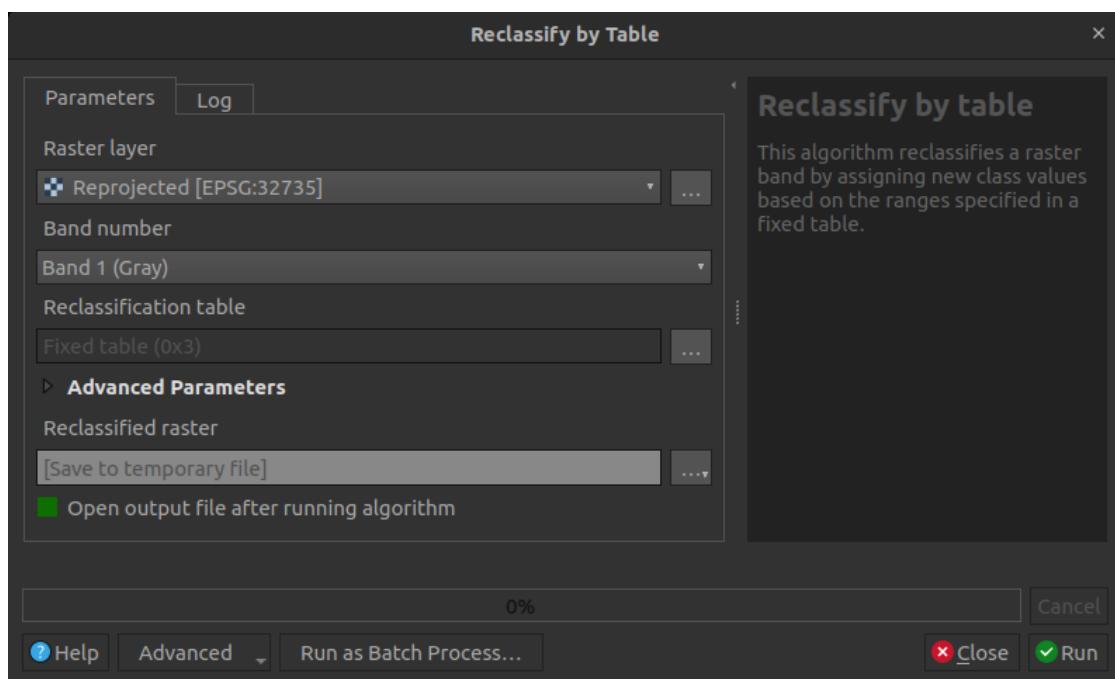


Figure 10: QGIS Reclassify by Table tool

- Open the Reclassification table so that the user can set the value reclassifications. Atleast one row needs to be provided
- Click **Add Row**
- Provide a Minimum and Maximum value. Consider the following:
 - Minimum must be less than maximum if providing a range
 - If only a single value needs to be reclassified, set the Minimum and Maximum to the same value
- Set the new Value for each row:
 - Value has to be numeric
 - If a user wants to remove a pixel/value from the analysis, the value needs to be set to the nodata value of the raster (e.g. -9999). See the above section on the **Nodata value** on how to find the nodata value of a raster

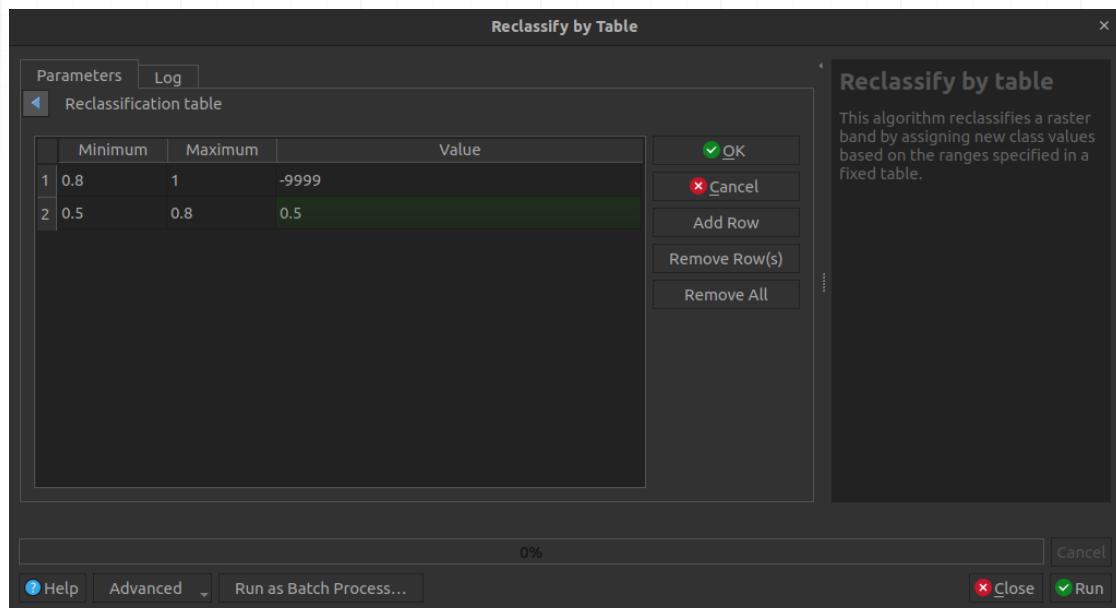


Figure 11: Reclassify table

- Click **Run**
- An example of the resulting raster compared to the original raster is shown in **Figure 12**

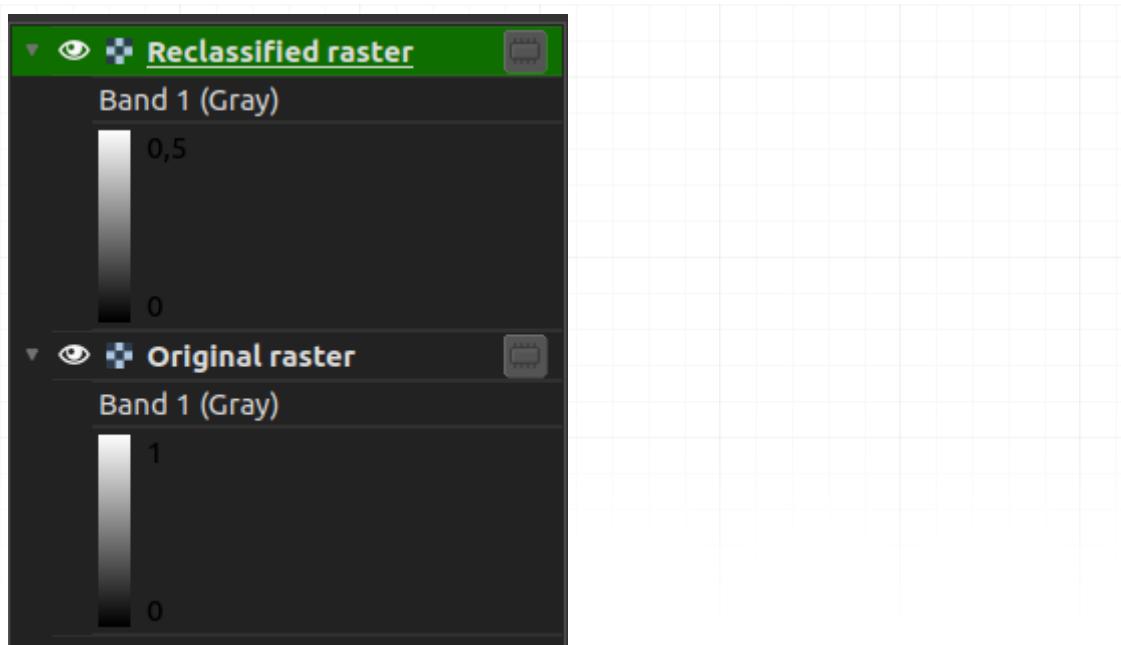


Figure 12: Reclassified raster compared to the original raster

Vector to raster

As mentioned above, the plugin can only work with raster layers. But often a user might have some data in vector format. This can easily be resolved by converting the vector layer to a raster, which can then be used as input to the plugin. Firstly, we want to convert the vector layer to make use of the same projected coordinate system than other data. This can be done as follows:

- Type 'Reproject layer' in the QGIS toolbox search
- Select the 'Reproject layer' tool in the 'Vector general' section



Figure 13: QGIS Reproject tool for vector layers

- Set the parameters as follows:
 - **Input layer:** Vector layer which needs to be reprojected
 - **Target CRS:** Coordinate system to which the layer should be reprojected, preferably a projected coordinate system
 - **Reprojected:** The output layer
- Click **Run**

Now that the vector layer is in the correct coordinate system, the user can convert the vector layer to a raster:

- Type 'rasterize' in the QGIS toolbox search
- Select 'Rasterize (vector to raster)'

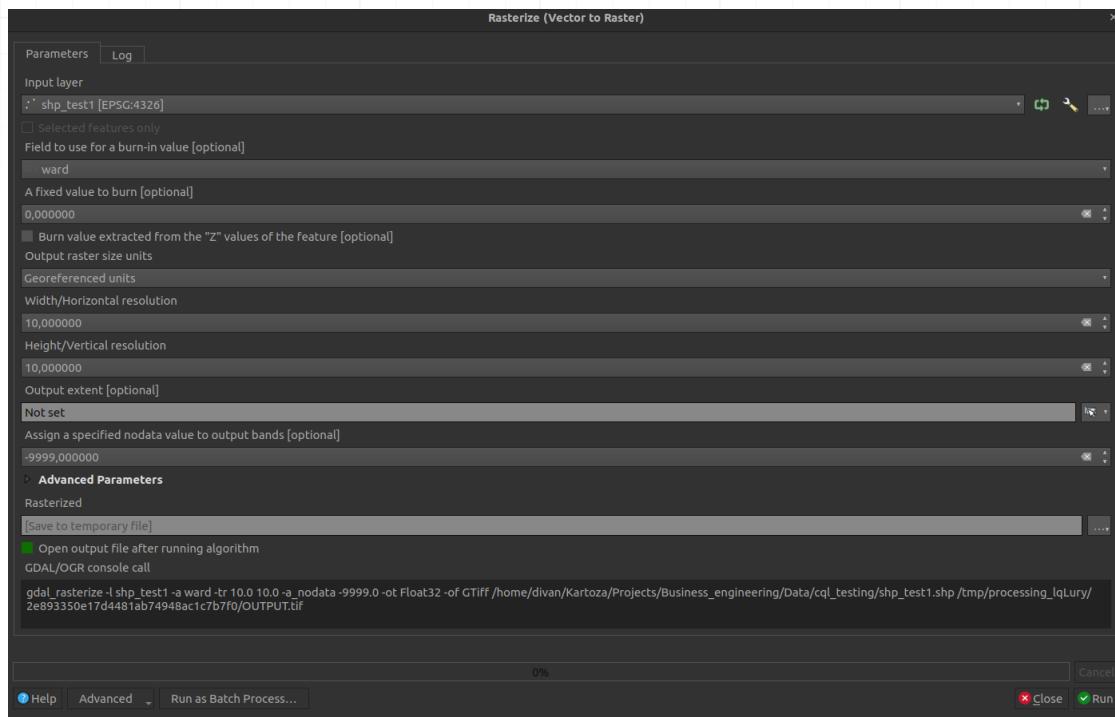


Figure 14: QGIS Rasterize tool

- Set the parameters as follows:
 - **Input layer:** The vector layer to convert to a raster
 - **Field to use to burn:** Attribute field to use as the raster pixel values
 - **A fixed value to burn:** A default value for empty fields for a feature. Otherwise leave as is
 - **Output raster size units:** Georeferenced units
 - **Width and Height:** Spatial resolution in meters. If the vector layer is in geographic coordinates, this distance will be degrees not meters
 - **Output extent:** Leave as is, except if the user wants to limit the output to an extent
 - **Assign a specific nodata value to output bands:** -9999 will suffice for most cases
 - **Rasterized:** The output raster
- Click **Run**

The user's data should now be ready for analysis.

1.3.2 CPLUS settings

The user can follow two approaches to open the CPLUS settings.

QGIS options (**Figure 15**):

- Click on **Settings -> Options**

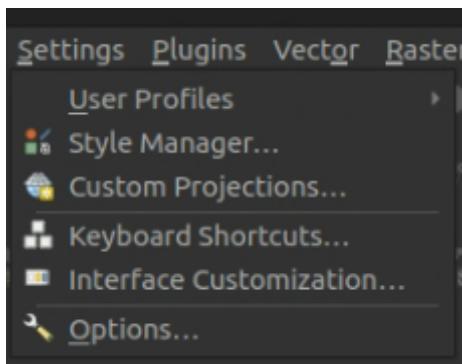


Figure 15: QGIS settings

- Select the CPLUS tab to the left
- This will open the CPLUS settings dialog. See **Figure 16** for an example

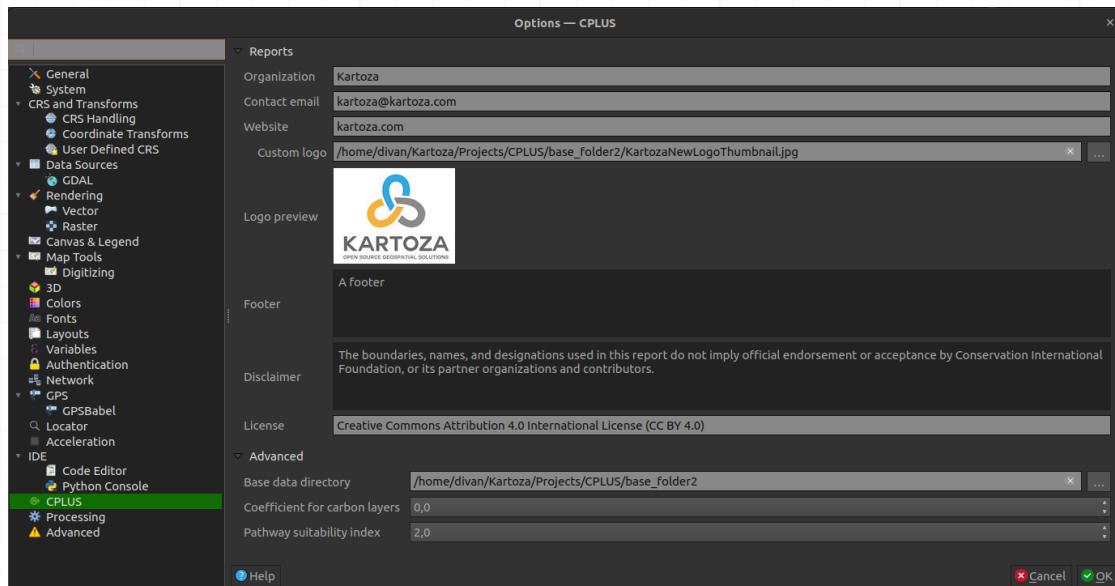


Figure 16: CPLUS section as loaded in the QGIS settings dialog

CPLUS toolbar (**Figure 17**): - Click on the CPLUS toolbar drop-down - Select **Settings** - This will take you directly to the CPLUS settings dialog in the QGIS options

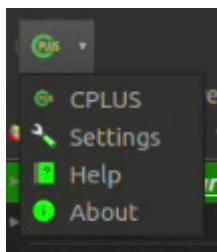


Figure 17: CPLUS toolbar button

A short description of each available setting a user can change. Most are optional, but the user needs to set the base directory as it's a requirement for the processing to work (e.g. outputs are stored in the base directory). Another important option to consider is snapping, as it will improve analysis results.

Configure Analysis:

- Settings will be added as the plugin development continues

Reports:

- Information which will be included when a report is generated. These settings are optional, and will be excluded from the report if not provided
- **Organization:** The organization(s) to be included in the report
- **Contact Email:** Contact email for the author
- **Website:** A website link to the project or company
- **Custom logo:** Enable and provide a custom logo of your choosing. If disabled, the CI logo will be used in the report
- **Footer:** Footer section for the report
- **Disclaimer:** A disclaimer to be added to the report
- **License:** A license to be added to the report

Advanced:

- **Base data directory** (required): Data accessed and download by the plugin will be stored here
- **Coefficient for carbon layers:** Value applied during processing to the carbon-based layers. Default is 0
- **Pathway suitability index:** Index multiplied to the pathways. Lower values means the pathway is less important, higher means its more important
- **Snapping:** Will set rasters to match the cell alignment of a reference layer
 - **Resample method:** Resampling performed on pixel values
 - **Reference layer:** The reference layer to which the cell alignment will be applied
 - **Rescale values:** Rescale values according to cell size

Figure 18 shows an example of updating and applying CPLUS settings.

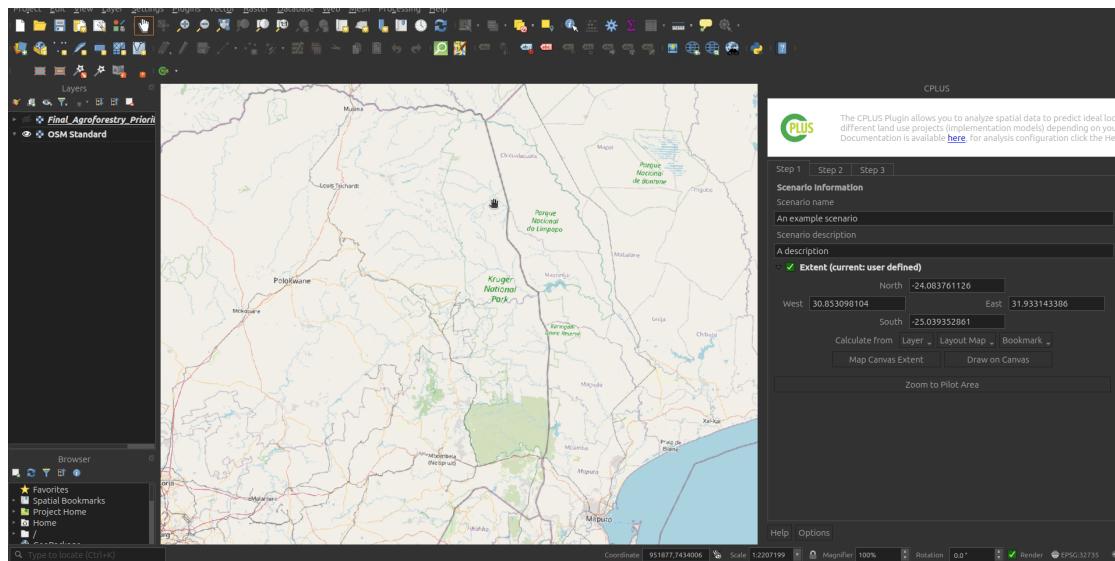


Figure 18: CPLUS settings example

1.3.3 Perform analysis

Figure 19 shows the toolbar button/menu for the plugin. Clicking on the icon will open the plugin.

When a user clicks on the drop-down button, they will be presented with four options:

- **CPLUS:** Close or open the plugin dock widget
- **Settings:** Open the settings for the plugin
- **Help:** Takes the user to the online guide for the plugin
- **About:** Will take the user to the About section on the GH pages

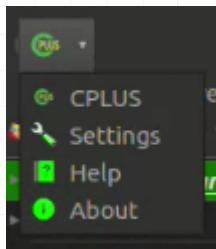


Figure 19: CPLUS toolbar icon

Open the CPLUS dockwidget by clicking on the CPLUS toolbar icon (**Figure 19**).

Step 1: Scenario Information

The first step focuses on the **Scenario Information**. A Scenario refers to an overall analysis done in an area of interest (AOI). Different criteria and priorities for spatial decision-making and comparison will be considered for each scenario.

- **Scenario name:** A name for the analysis to be performed
- **Scenario description:** A detailed description of the analysis
- **Extent:** The area of interest for this analysis. This can be calculated from the current canvas view extent, a layer, or an extent drawn by the user
- **Figure 20** shows an example of Step 1
- Once the information has been provided, click **Step 2**

Note:

If the QGIS canvas CRS is not set to WGS84 (EPSG: 4326), the zoom to pilot area will not happen.

Step 1 Step 2 Step 3

Scenario Information

Scenario name
An example Scenario

Scenario description
This is only an example

Extent (current: map view)

North: -23.949302012
West: 30.194845223
East: 32.191118789
South: -25.534328685

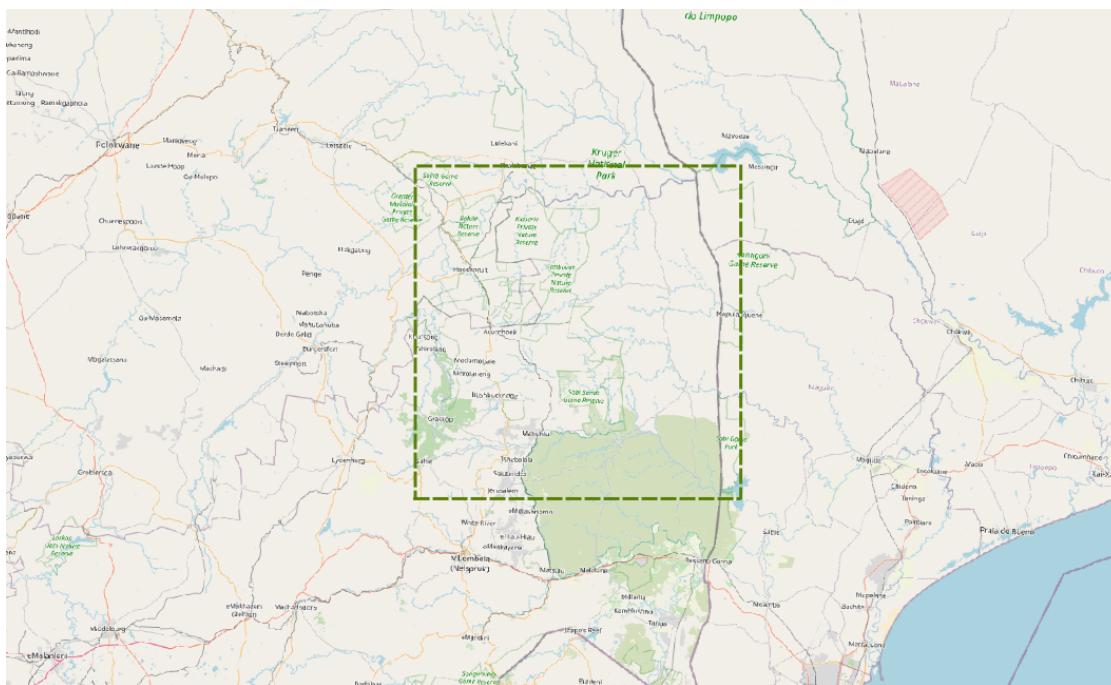
Calculate from Layer Layout Map Bookmark
Map Canvas Extent Draw on Canvas

Zoom to Pilot Area

Figure 20: Step 1 focuses on Scenario Information

Pilot area

The pilot study area covers Bushback Ridge, South Africa. When a user's study area is outside of this region, some of the Implementation models and Priority weighted layers will be disabled. This is because those datasets are specific to the Bushback Ridge study area and are of no use for other AOIs. It's important for a user to take this into account, as step 2 and step 3 will be affected by this.



If the selected extent is outside of this region, the Bushback Ridge Implementation models will be disabled.

The screenshot shows the CPLUS interface divided into two main sections:

- NCS Pathways:** This section lists various land management and restoration activities. Some items have a slash followed by a sub-item, indicating a relationship or a specific action. The listed items include:
 - Agroforestry
 - / Alien Plant Removal
 - Animal Management
 - Avoided Deforestation
 - Avoided Grassland Conversion
 - Avoided Savanna Woodland Conversion
 - Avoided Wetland Conversion
 - Fire Management
 - Restoration - Forest
 - Restoration - Savanna
 - Restoration - Wetland
 - / Sustainable Agriculture Crop Farming
 - Woody Encroachment Control
 - agroforestry
 - avoided open woodland
- Implementation Models:** This section lists various models or practices, many of which correspond to the items in the NCS Pathways section. Some items are bolded, suggesting they are active or recommended. The listed items include:
 - Agroforestry**
 - Alien Plant Removal**
 - Applied Nucleation**
 - Assisted Natural Regeneration**
 - Avoided Deforestation and Degradation**
 - Avoided Wetland Conversion/Restoration**
 - BioProducts**
 - Bush Thinning**
 - Restoration - Savanna
 - Direct Tree Seeding**
 - Fire Management
 - Livestock Market Access**
 - Livestock Rangeland Management**
 - Natural Woodland Livestock Management**
 - Protected Area Expansion**
 - Sustainable Crop Farming & Aquaculture**
 - agroforestry123**
 - agroforestry
 - woodland**
 - avoided open woodland

Each section has a set of small icons at the bottom, likely for filtering or selecting specific layers. Below each section is a "Description" input field with a text area and a "Save" button.

The same goes for the Priority Weighted layers.

Weighting priorities

Assign priority weights using the sliders. Add the priority weighting layers unique to each implementation model by clicking and dragging the layer under a group title.

Priority groups

- Climate R** Lc Hi
- Finance -** Lc Hi
- Finance -** Lc Hi
- Livelihood** Lc Hi
- Policy** Lov Hi
- Ecological** Lc Hi
- Finance -** Lc Hi

Priority Weighting Layers

- biocombine_clip_norm
- cccombo_clip_norm
- cccombo_clip_norm_inverse
- ei_all_gknp_clip_norm
- Policy
- social_int_clip_norm
- social_int_clip_norm_inverse

Run Scenario

If a user is outside the Bushback Ridge region, they will need to create custom IMs and/or PWLs.

Explanation on these follows in the following sections.

Step 2: Pathways and models

This step deals with the **Natural Climate Solution (NCS) pathways** and the **Implementation models (IM)**. A NCS pathway can be defined as a composite spatial layer on specific land use classes and other factors that determine areas ideal for a specific use case (e.g. Animal management). An IM is a combination of NCS pathways represented in an AOI spatial layer. **Figure 21** shows the UI.

The screenshot shows the 'Step 2' tab selected in the top navigation bar. On the left, under 'NCS Pathways', the 'Avoided Wetland Conversion' item is highlighted with a green background. On the right, under 'Implementation Models', the 'Bush Thinning' item is also highlighted with a green background. Both sections have a list of items with small green arrows pointing right next to them. Below each list is a row of four small icons: a green plus sign, a pencil, an equals sign, and a circular arrow.

Description	
Avoids carbon emissions by preventing the con	Bush thinning refers to the controlled removal

Figure 22: Step 2 allows the user to create and edit NCS pathways and Implementation Models

Step 2 buttons (**Figure 18**):

- Add:** Adds a new pathway or model
- Delete:** Delete a pathway or model
- Editing:** Edit an existing pathway or model
- Refresh view:** Checks the base directory for data



Figure 23: Create, delete, and edit buttons

NCS Pathway

-
- Click on the left green plus button to add a new pathway (**Figure 24**)
 - Provide a **Name** and **Description** for the pathway
 - Two approaches to select a layer: A layer from the **QGIS canvas**, or **Upload from a file**
 - Add **Carbon layers** as desired. Multiple carbon layers can be provided. These layers will be averaged
 - Click **OK**

- The new **NCS pathway** will be added

 **Note:**

If the NCS pathway is broken (e.g. layer or file cannot be found), the pathway text will have an exclamation mark next to it. The user will need to rectify the issue before continuing to step 3.

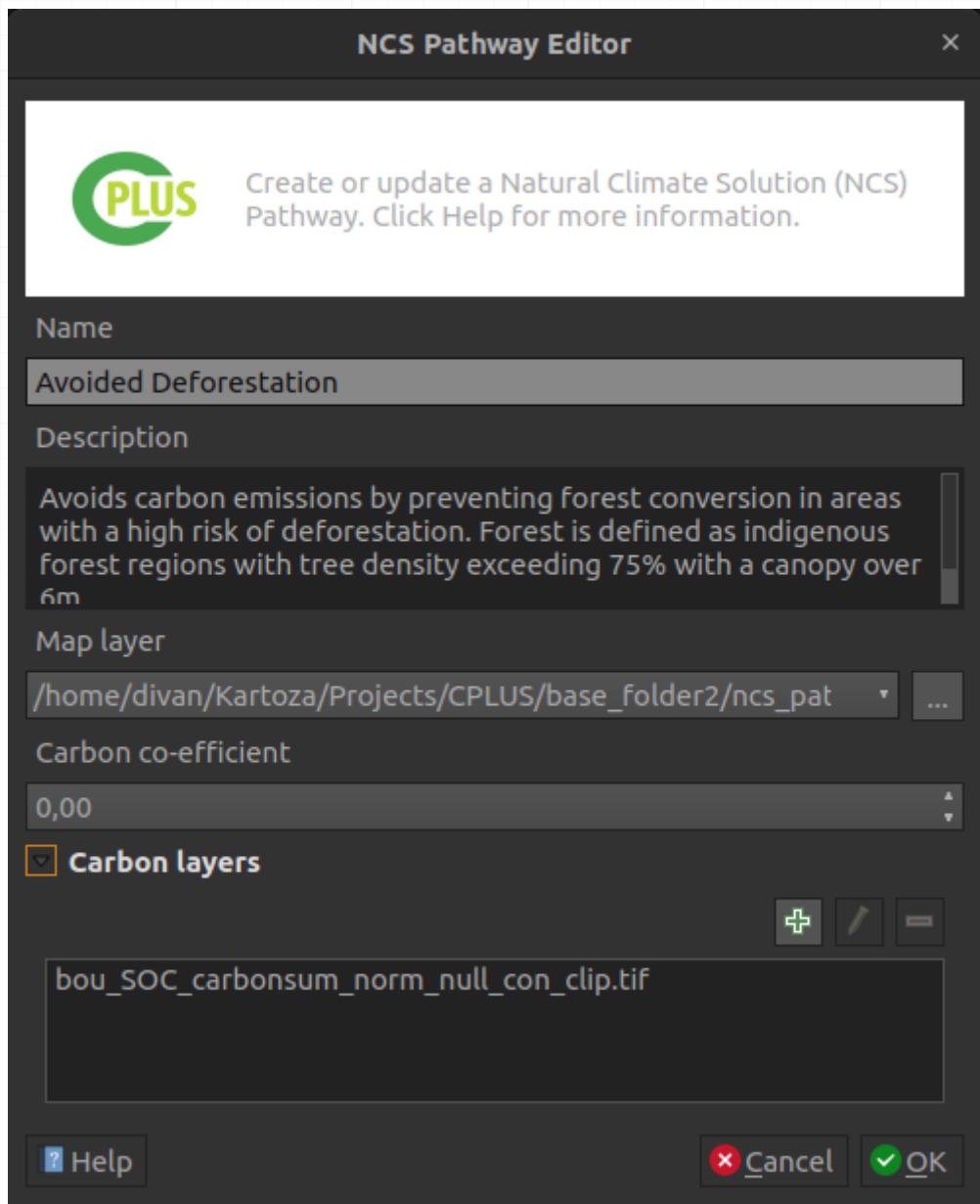
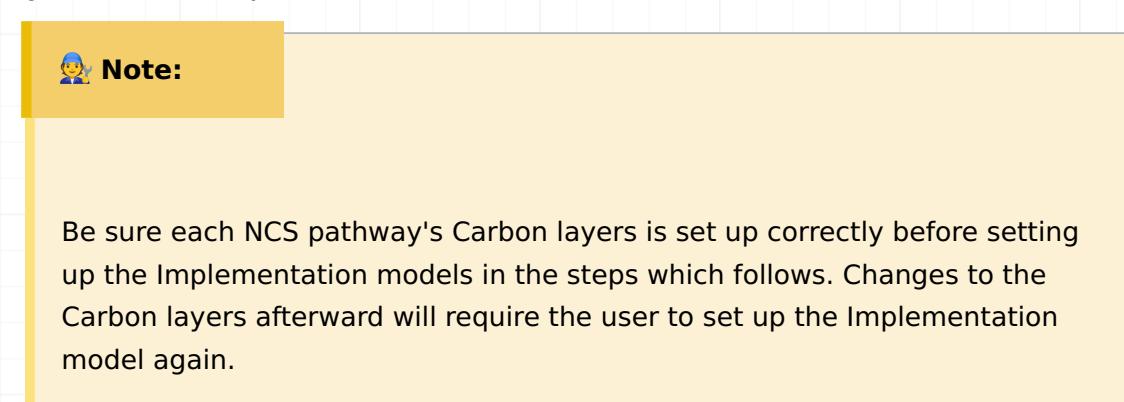


Figure 24: NCS Pathway creator/editor



Add pathways to an existing IM:

- Select the IM to which a pathway should be added
- Select the pathway you want to add to the IM
- Click the right arrow ➤ to add a pathway to the selected IM
- Click the double right arrow ➡➡ to add all pathways to the IM
- The user can also drag-and-drop a pathway onto the desired IM

How to add a new IM:

- Click on the right green plus button to add an **Implementation model** (**Figure 25**)
- Provide a **Name** and **Description**
- (optional) The user can provide an existing raster for the IM. Enable **Map layer** to do this
- Click **OK**
- The new **Implementation model** will be added

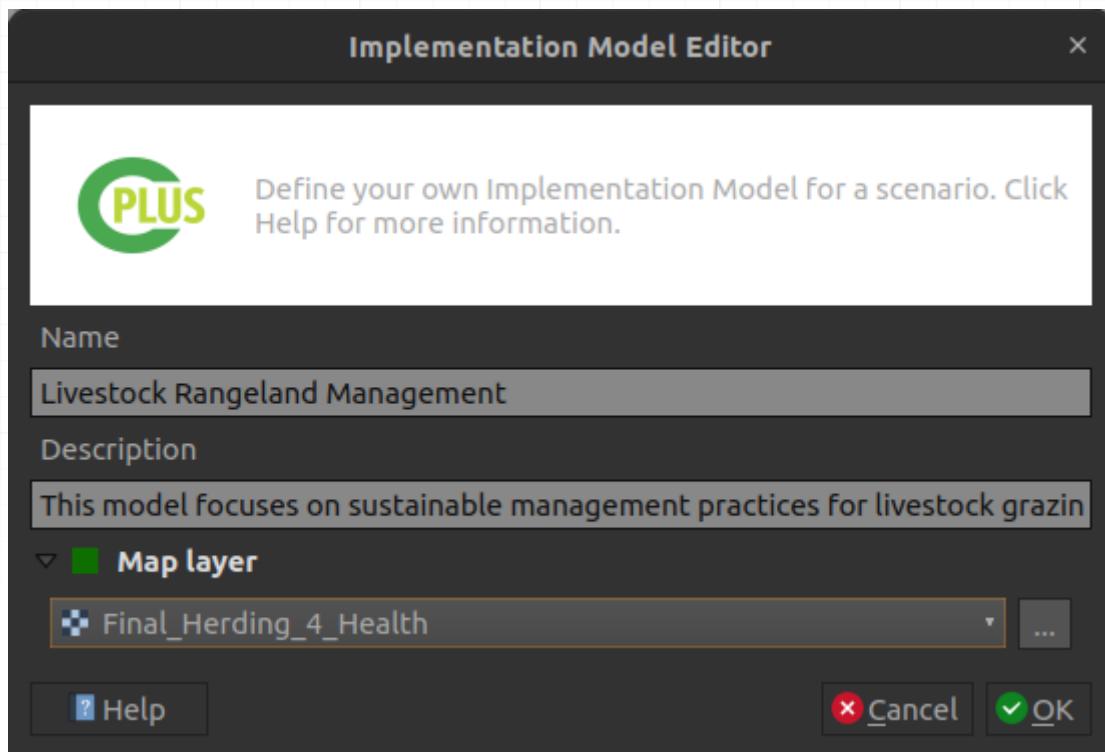
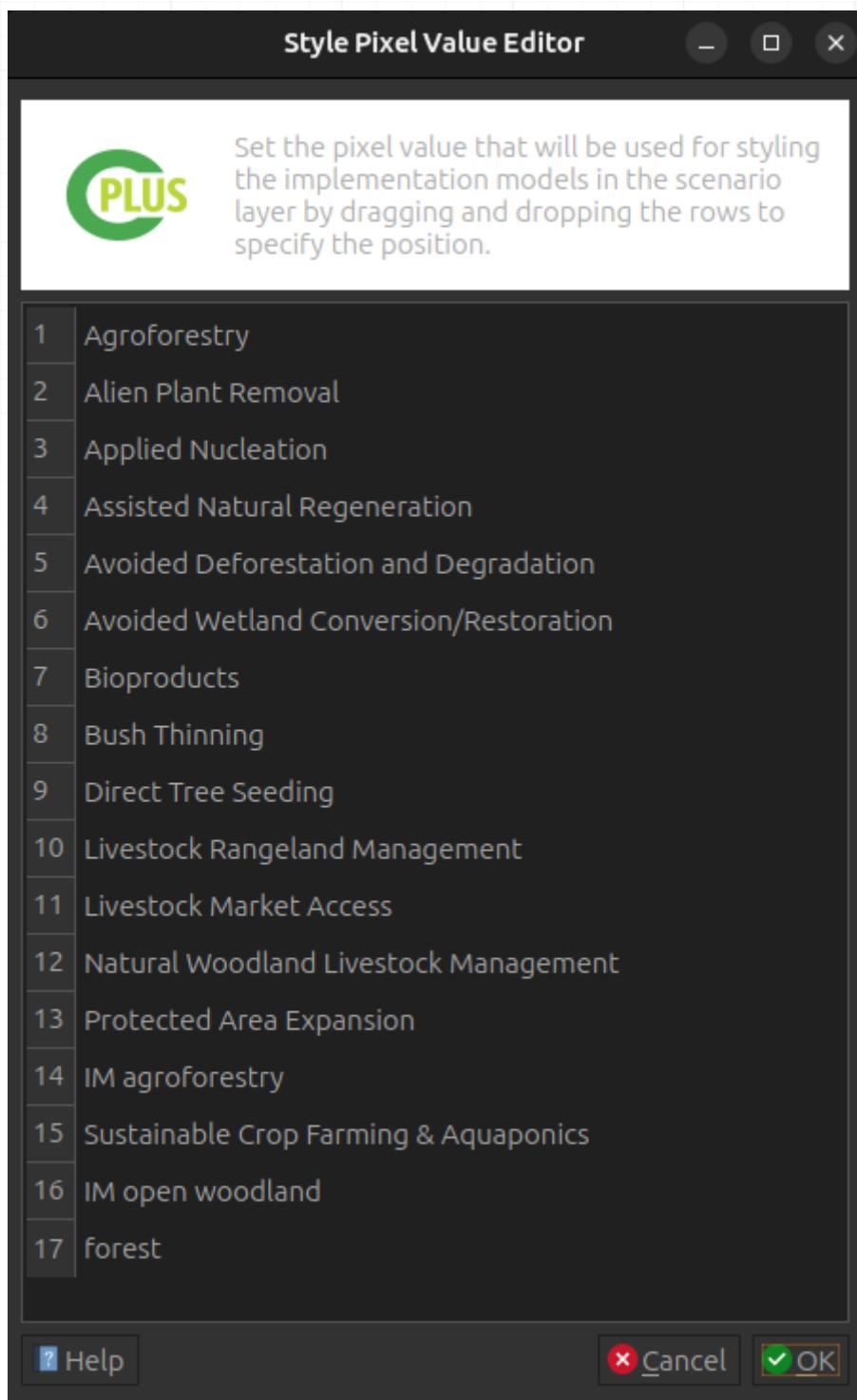


Figure 25: Implementation Model creator/editor

- Open the Style pixel value editor by clicking on the  button
- Select the IM which needs to be moved up or down in the stack
- Drag-and-drop the IM where it needs to be in the stack
- Click **OK** once done



- The final step is to select each of the IMs a user want to include in the scenario run
- A user can exclude IMs if they don't want to include it, even if the IM has pathways

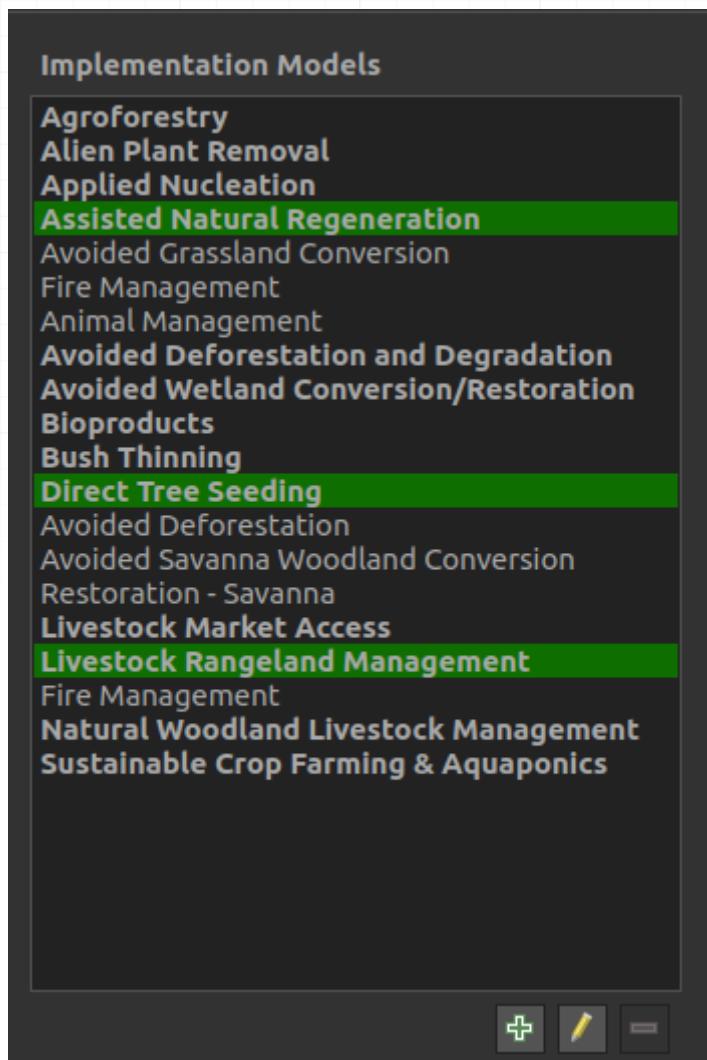


Figure 26: Selected Implementation models

 **Note:**

Before proceeding to Step 3, a user needs to define at least one NCS pathway layer for an implementation model, otherwise a warning message will be displayed.

Step 3: Priority weighting

The final step deals with the **Weighting priorities** and **Priority groups**. These weights will be applied when the user starts running the scenario. An example is shown in **Figure 27**.

- Weight values ranges from 0 to 5, and affects how important a PWL is compared to other layers
- A value of 0 indicates that the PWL has a lower importance
- A value of 5 means that the PWL has a higher importance

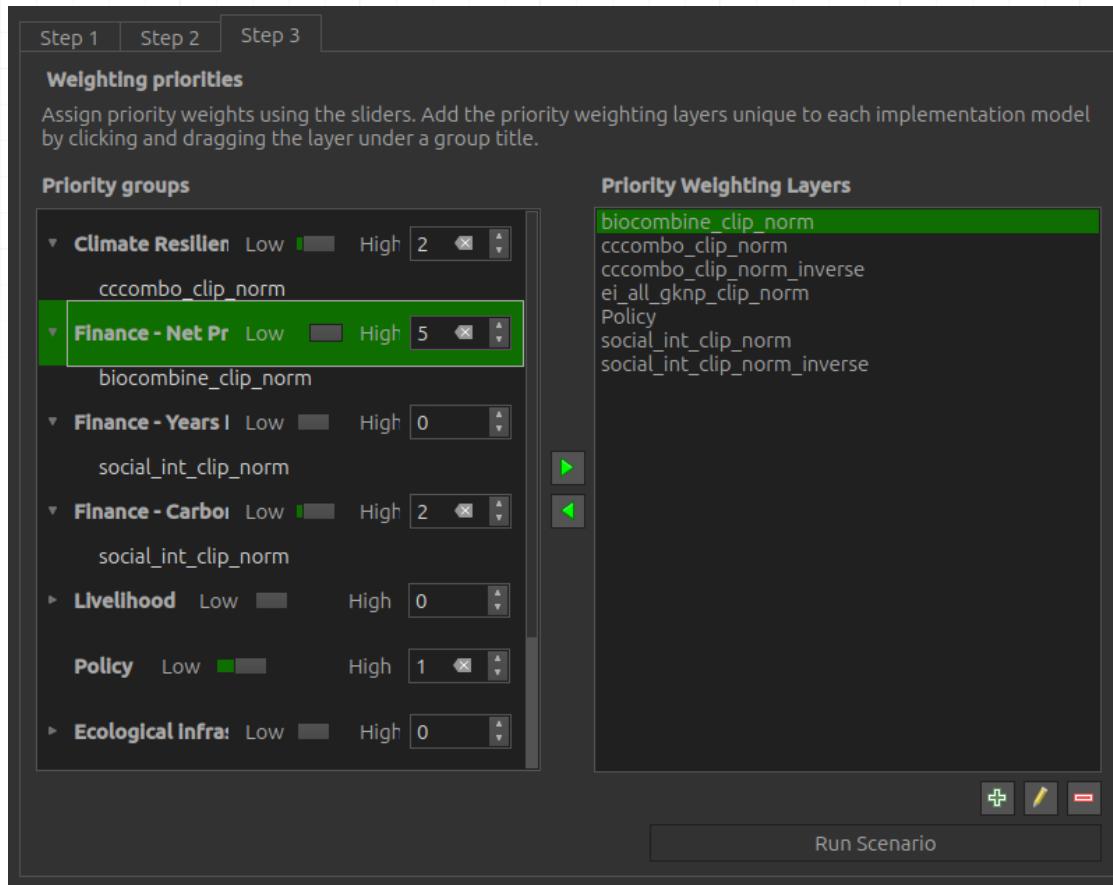


Figure 27: Step 3 allows the user to set the Weights of each Priority Group

The priority weighting layers can be selected and added and removed into each priority group by using the arrow buttons.

Add priority layers

Select the target layer from the priority weighting layers list and the destination group from the priority groups and use the left arrow button to add the layer into the group.

Remove priority layers

Select the target layer from the priority weighting layers list from its priority group and use the right arrow button to remove the layer into the group.

Create custom priority layers

- Click on to add a new custom priority layer, or to edit an existing priority layer

- This will open the Priority Layer dialog (see **Figure 23**)
- The following parameters needs to be set:
 - **Priority layer:** The layer which represents the priority layer
 - **Priority layer name:** A unique identifier for the priority layer
 - **Priority layer description:** A detailed description of the priority layer
- Click the **Assign implementation models** button to select IMs to be associated with the priority layer (see **Figure 28**)

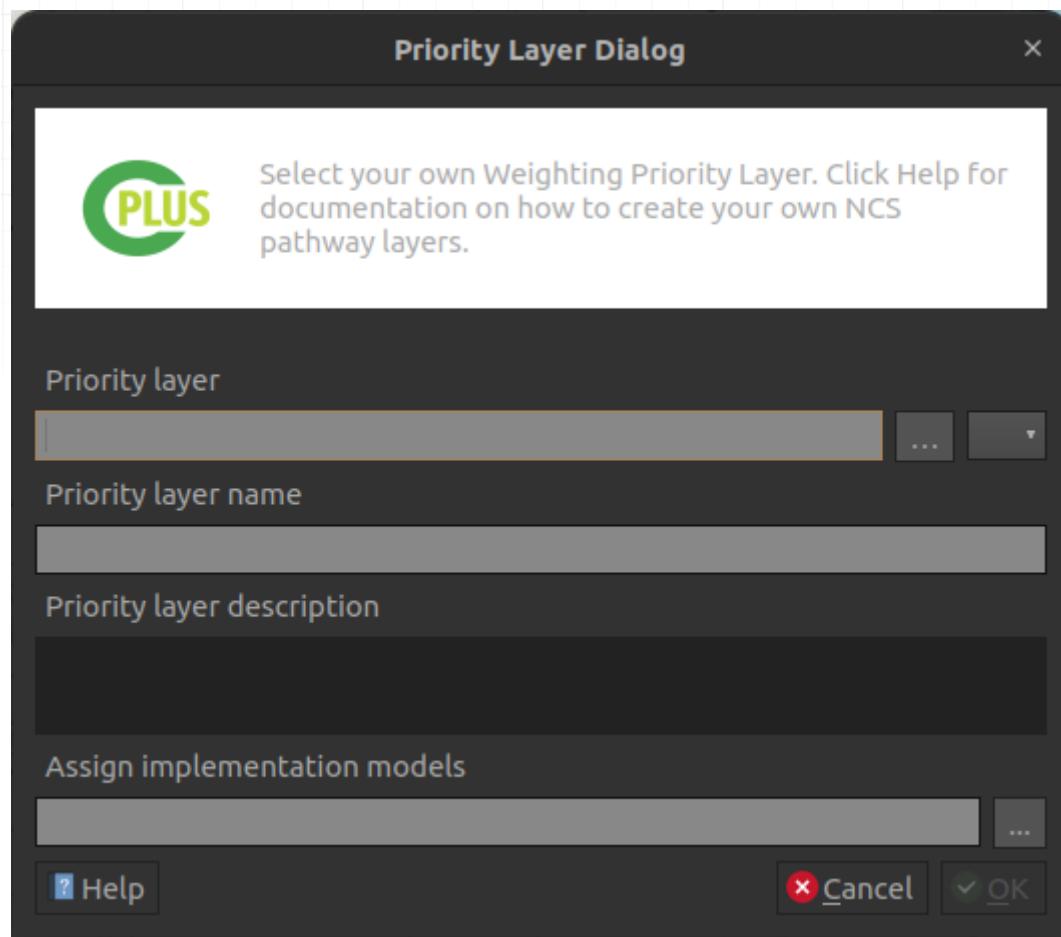


Figure 28: Priority layer dialog

- Select the IMs you want to be associated with the priority layer
- Click **OK**

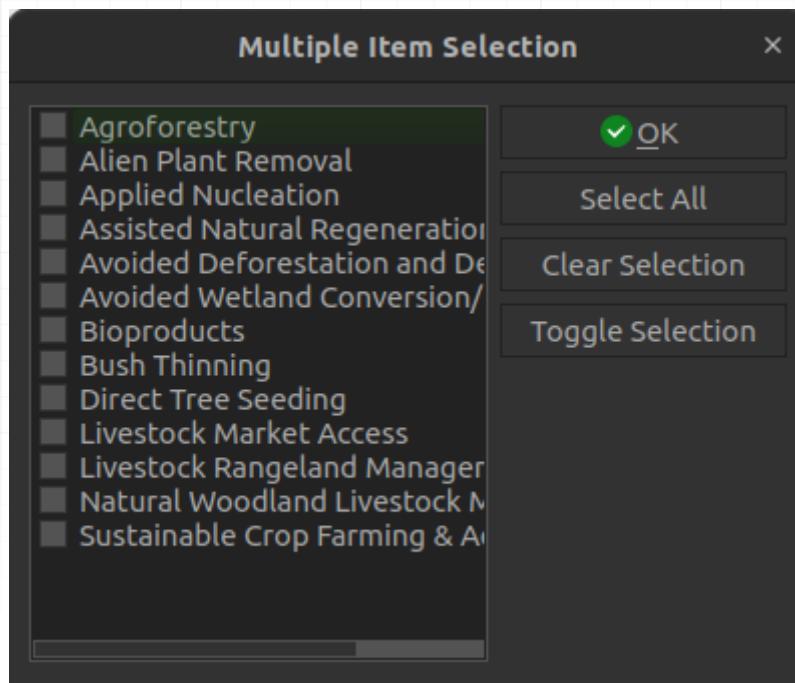


Figure 29: Implementation model selection for priority layers

- Remove the selected PWL

Setting groups values

Move the slider to adjust the weight of each group, values can also be set manually, by using the left input spin box. Once done selecting weights, click **Run Scenario** button to run the analysis.

Steps 1 to 3 example

The following recording (**Figure 25**) shows an example on how to do Step 1, 2 and 3. This is based on the pilot study area.

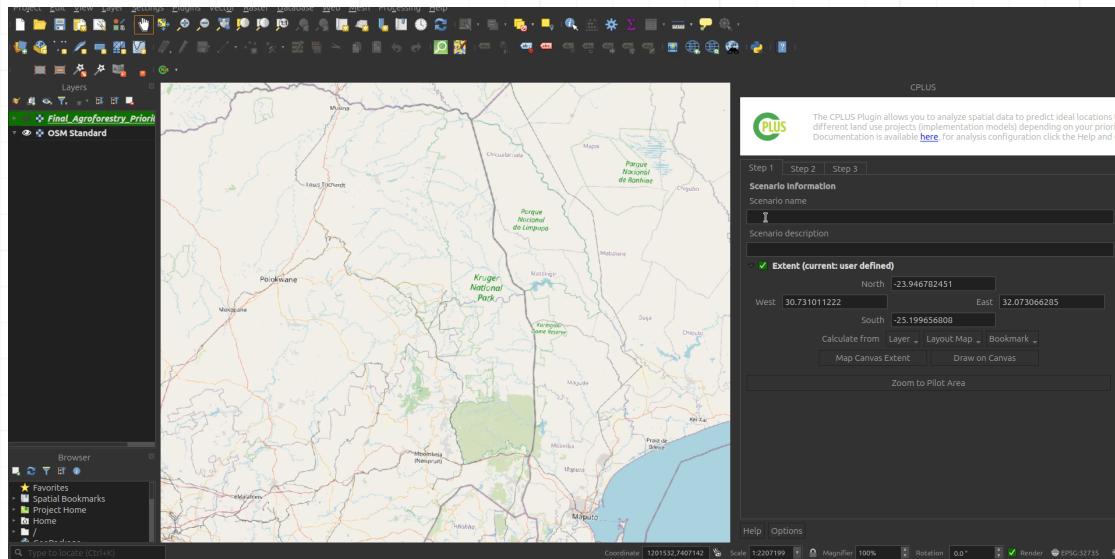


Figure 30: Shows how to implement Step 1, 2 and 3 in QGIS

1.3.4 Processing

- Once the user has provided all desired parameters, click **Run Scenario**
- The processing dialog will open (**Figure 31**)
- The processing will take a while, depending on the number of IMs and pathways provided for each IM
- Click the **Cancel** button to stop the processing

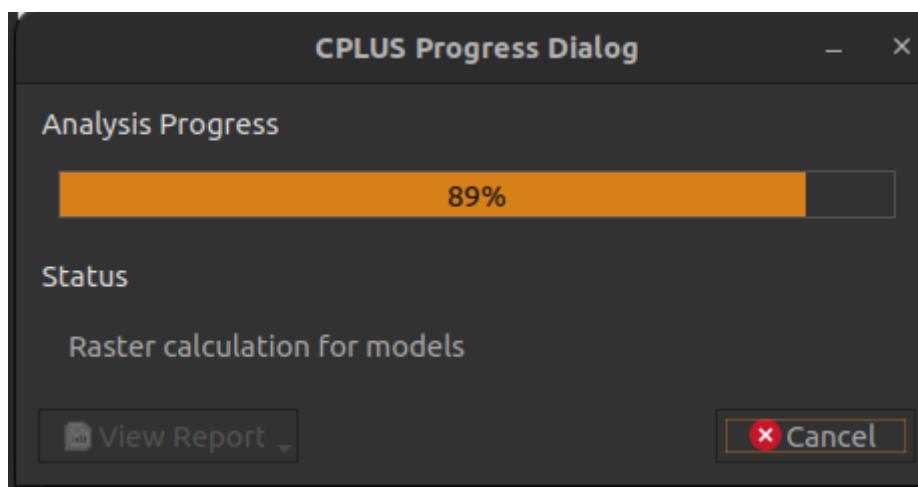


Figure 31: Processing dialog while the algorithm is running

- Figure 32** will be the result if the processing succeeded
- The user should take note that the **View Report** button is now available

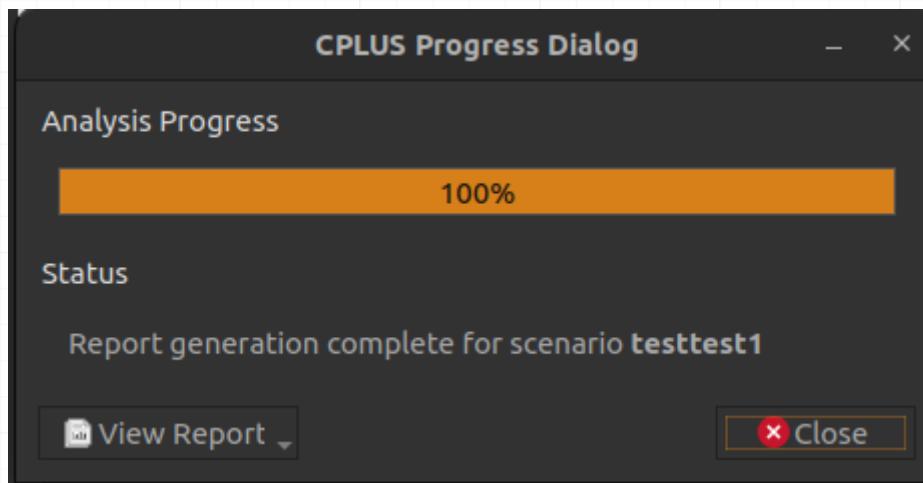


Figure 32: Processing dialog if successfull

Processing results

The following groups and layers will be added to the QGIS canvas once the processing finishes (see **Figure 33**): - A group containing the Scenario results - **Implementation Model Maps**: Non-weighted IMs created by the user in Step 2 - **Weighted Implementation Model Maps**: Weighted IMs based on the IMs added in Step 2 and weighing set in Step 3 - **NCS Pathways Maps**: Pathways used for each IM in Step 2. If a IM layer were provided as the IM in Step 2, this will contain no pathways

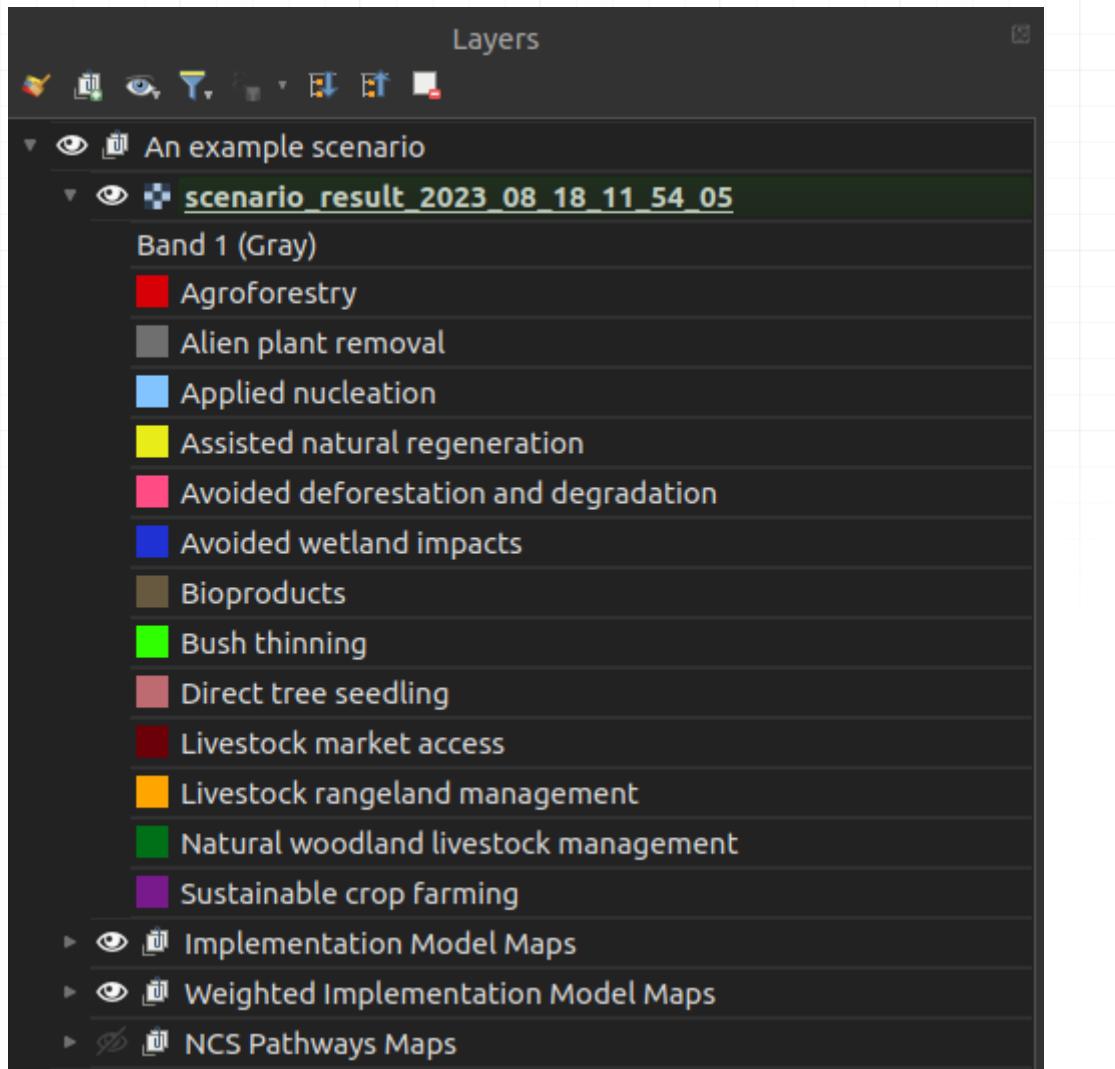


Figure 33: Groups and layers added to the QGIS canvas

An example of output results in QGIS is detailed by **Figure 33**

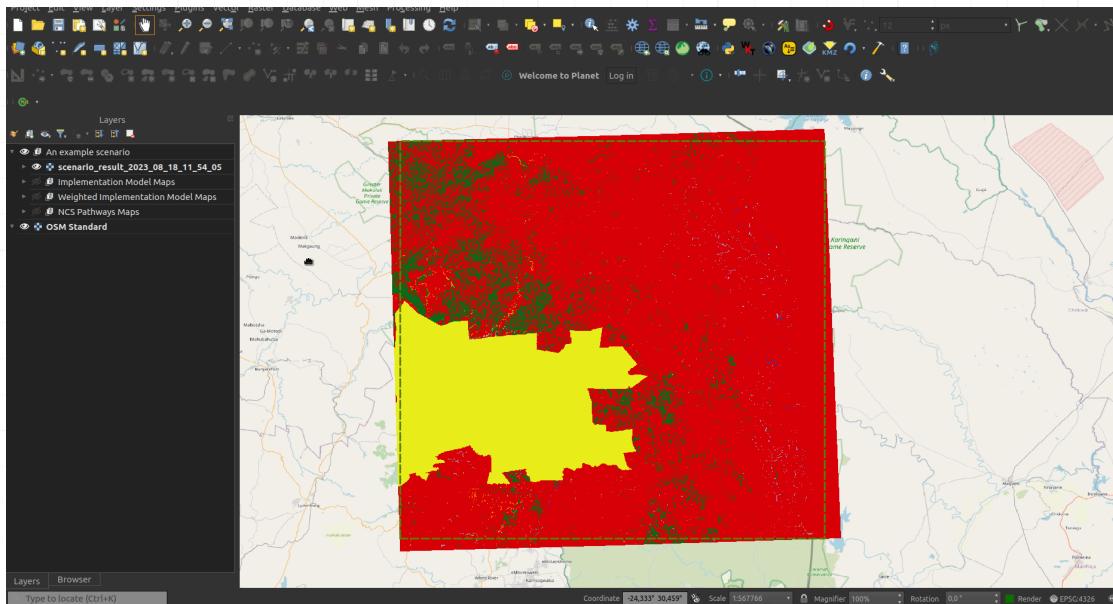


Figure 34: A recording example of an example scenario

1.3.5 Report generating

- Click the **View Report** button
- The user will have the following options:
 - **Layout designer:** Opens the report in the QGIS layout designer
 - **Open PDF:** Opens the report in PDF format
 - **Help:** Open the help documentation related to the reports

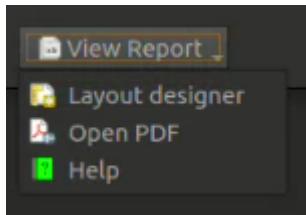


Figure 35: Report options

- **Figure 35** shows an example of a report opened in the layout designer

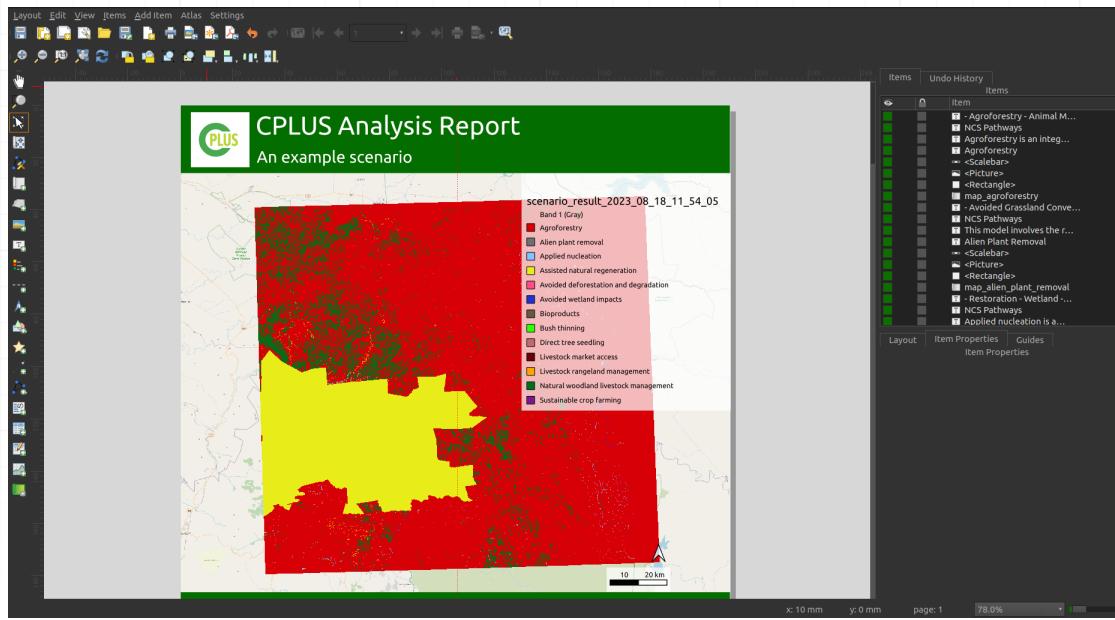


Figure 36: Report opened in the QGIS layout designer

- **Figure 36** shows a report in PDF format

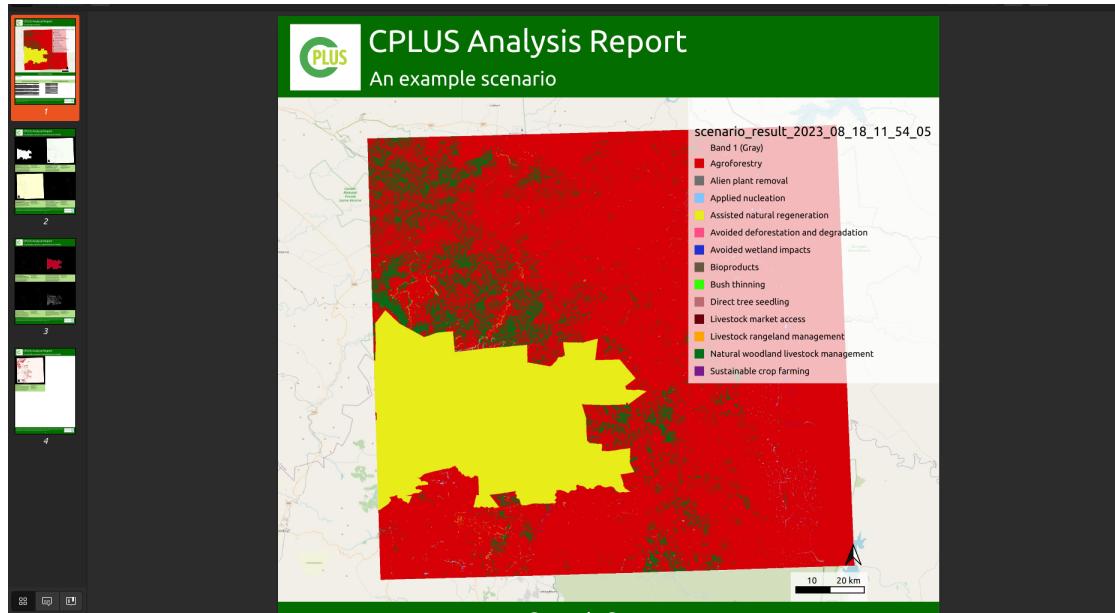


Figure 37: PDF version of a report

Generated report example

Here is an example on how to open a report in the QGIS layout designer, or as a PDF (**Figure 33**).

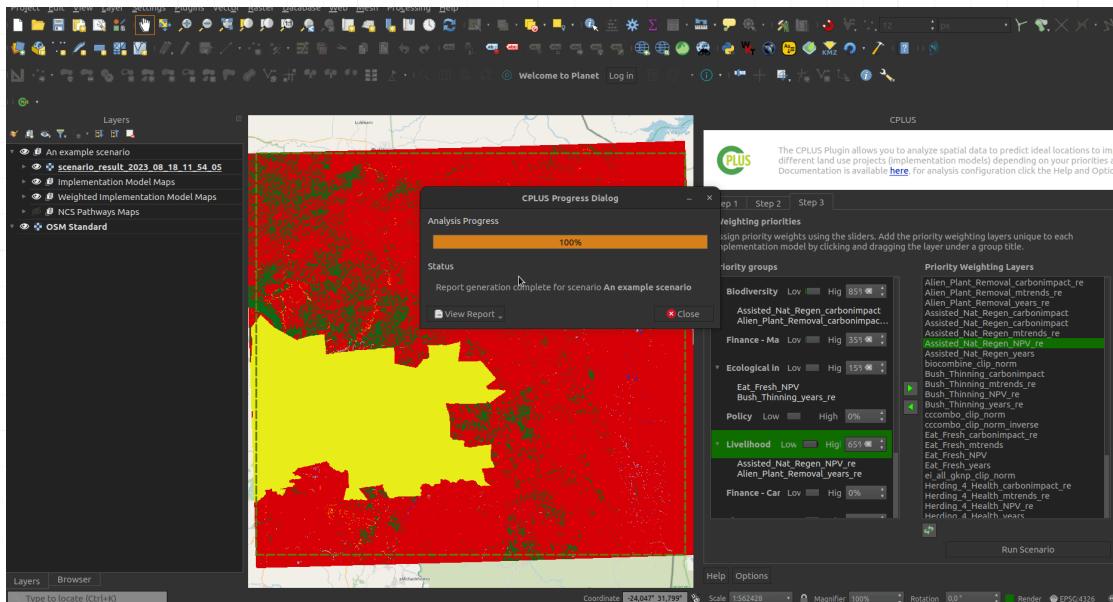


Figure 38: Example of a generated report in PDF and layout designer formats

1.4 Manual

The manual covers two sections. Firstly the workflow will be covered. This includes a discussion on the calculations and formulas. This is so that a user can understand how the CPLUS processing workflow and calculations for each step is done when processing the pathways and carbon layers, how the implementations models (IM) are created, algorithms applied to create the priority weighted layer (weighted IM), and the last step, which is the highest position calculation.

The second section deals with the plugin itself. It covers each step, explains each element of each step and why its needed. A description of the generated report are also provided.

1.4.1 CPLUS calculations and formulas

Figure 1 shows the workflow of the CPLUS model. The workflow can be split into four parts:

- Natural climate solution (NCS) weighted carbon pathway(s)
- Implementation model (IM)
- Priority weighted layer (Weighted IM)
- Highest position (Scenario result)

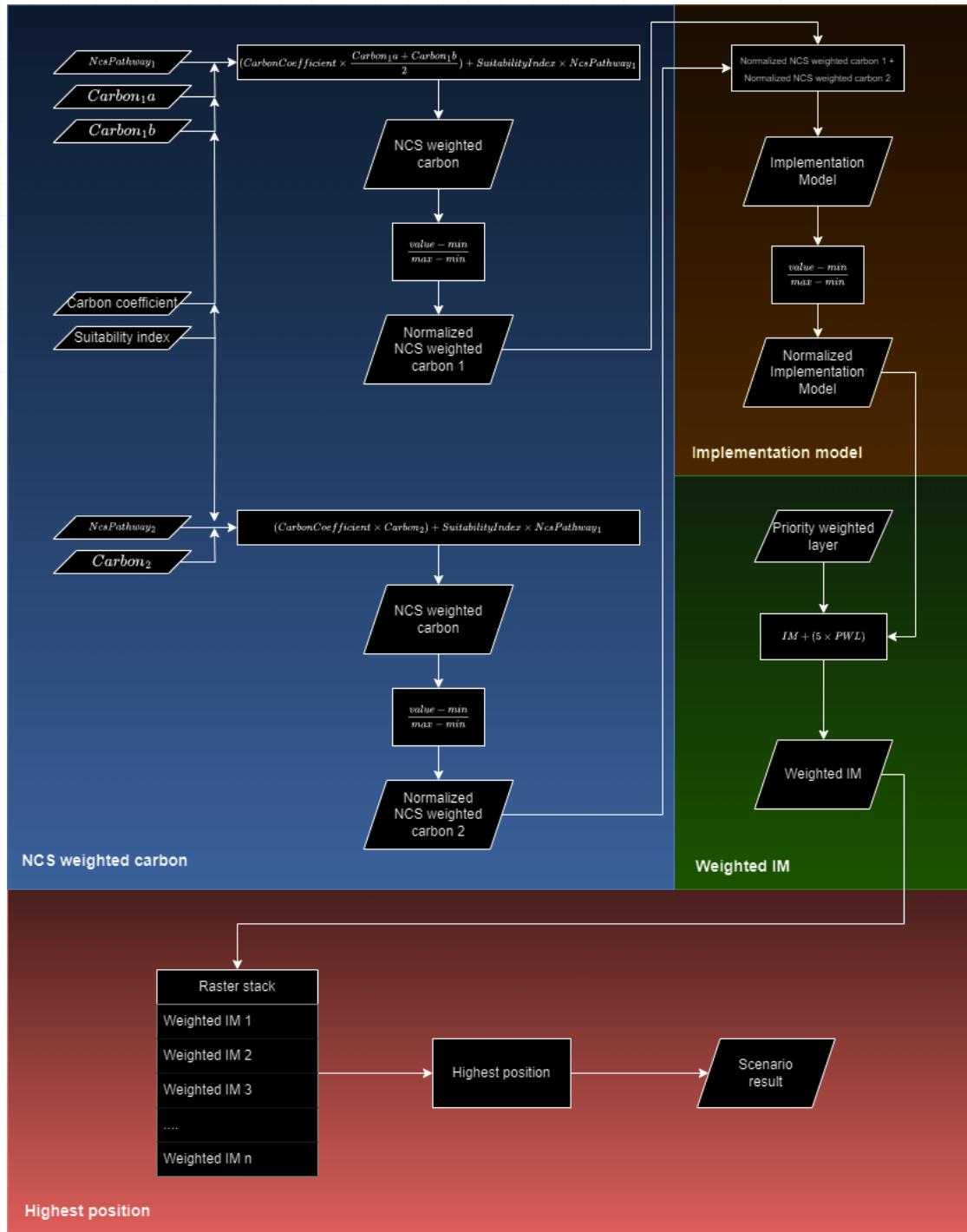


Figure 1: CPLUS workflow

NCS weighted carbon

The following steps/rules are considered to create the NCS weighted carbon layer(s):

- Carbon layers:
 - When multiple Carbon layers are provided, the average is calculated from the layers to create a single Carbon layer
 - The produced Carbon layer is multiplied by the Carbon coefficient provided by the user in the settings
 - If the Carbon coefficient is zero, the value is ignored
- NCS pathways:
 - Multiply the pathway raster with the Suitability index
 - If the index is zero, the pathway raster is used as-is
- **Equation 1** shows how the NCS weighted carbon layer is calculated

```
$$ \operatorname{NCS\ weighted\ carbon} = \{CarbonCoefficient\} \times \frac{(Carbon_1 + Carbon_2 + ... + Carbon_n)}{n} + (\{SuitabilityIndex\} \times NcsPathway) $$
```

Equation 1: NCS weighted carbon

where CarbonCoefficient is the carbon coefficient value multiplied with the averaged carbon raster;

Carbon is a carbon raster;

SuitabilityIndex is the NCS pathway index value;

NcsPathway is the NCS pathway raster; and

n is the number of carbon rasters.

- The results from the above calculation are normalized to create the normalized NCS Weighted Carbon layer
- A normalized raster's pixel values range from 0 to 1
- Normalization is done as shown in **Equation 2**

```
$$ \operatorname{Normalized\ NCS\ weighted\ carbon} = \frac{\operatorname{value} - \min}{\max - \min} $$
```

Equation 2: Normalized NCS weighted carbon

where value is the pixel value;

min is the minimum value of the raster; and

max is the maximum value of the raster.

Implementation model

-
- Because an IM can consist of multiple pathways, the normalized results will be summed
 - All NCS weighted carbon layers, as created from **Equation 2**, is summed as shown in **Equation 3** to create the IM from the pathways

```
$$ \operatorname{Summed\ pathways} = NcsWeightedCarbon_1 + NcsWeightedCarbon_2 + ... + NcsWeightedCarbon_n $$
```

Equation 3: Summed pathways for the IM

where NcsWeightedCarbon is a pathway set up by the user; and

n is the number of pathways.

- Now that the pathways has been summed for the IM, the result needs to be normalized
- The Suitability index and the Carbon coefficient then needs to be taken into account after the normalized raster has been created
- This calculation is shown in **Equation 4**

```
$$ \operatorname{FinalIM} = \{(\text{SuitabilityIndex} + \text{CarbonCoefficient})\} \times \frac{\text{value} - \text{min}}{\text{max} - \text{min}} $$
```

Equation 4: Final IM created from pathways

where value is the pixel value;

min is the minimum value of the raster;

max is the maximum value of the raster;

SuitabilityIndex is the NCS pathway index value; and

CarbonCoefficient is the carbon coefficient value multiplied with the averaged carbon raster.

- The resulting output is the final IM

Priority weighted layer (Weighted IM)

- This step is performed after the IMs has been created
- The PWL is more important, and will therefore be multiplied by five to take this into account
- The PWL weighted is calculated as shown in **Equation 5**

```
$$ \operatorname{PriorityWeightedLayer} = \{ \text{FinalImplementationModel} + (\{5\} \times \text{PriorityWeightedLayer}) \} $$
```

Equation 5: Priority weighted layer (Weighted IM) calculation

- The resulting PWL will then be used as input to the Highest position calculation

Highest Position

The [Highest position](#) tool determines the raster in a stack with the highest value at a given pixel. Essentially the result is a classification, where each class represents a specific IM. If multiple rasters has the highest pixel value at a given pixel, the first raster with that pixel value in the stack will be used. Figure 2 shows an example from the QGIS description of the Highest position tool.

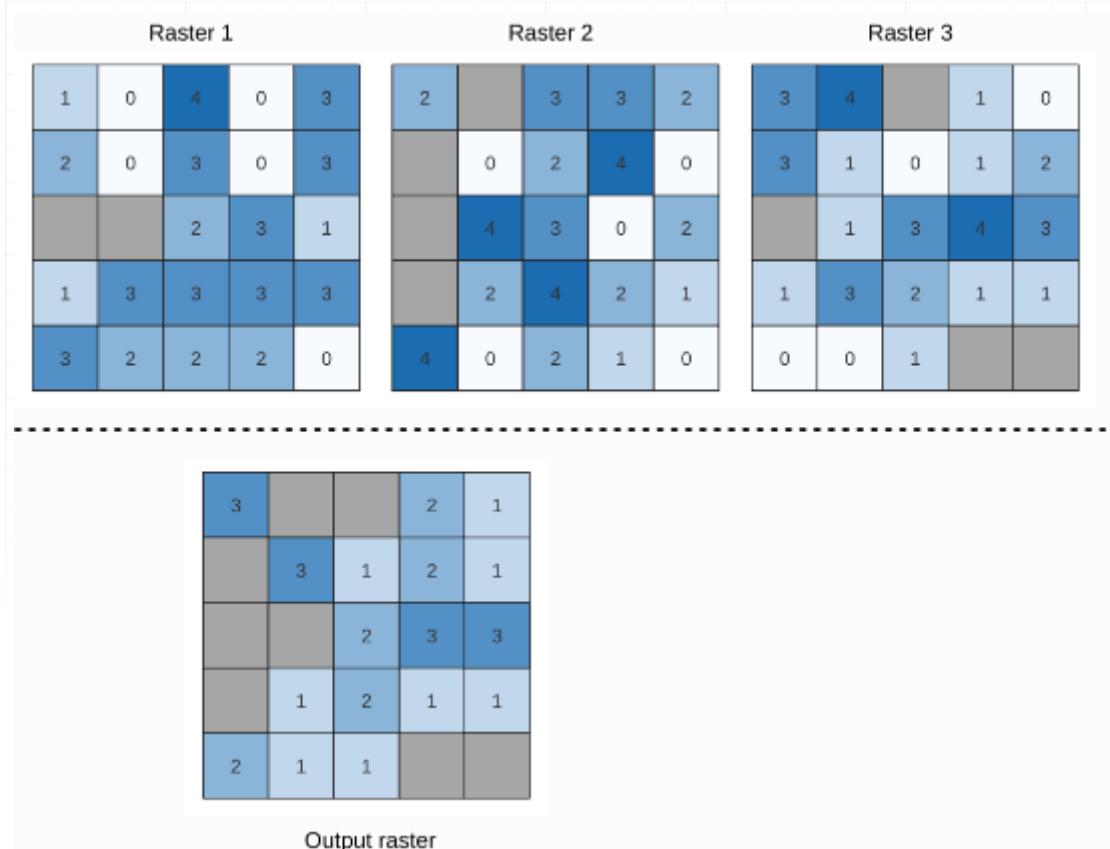


Figure 2: Highest position example

In the plugin the nodata values are ignored. This means that if atleast one raster has a pixel value at that cell there will be a raster stack value. If none of the rasters in the stack has a pixel value at that cell (e.g. each raster pixel is nodata) the output will be nodata at that pixel.

Here is an explanation on how-to use the **Highest position** tool:

- Figure 3 shows the layer for the Highest position at stack position 1

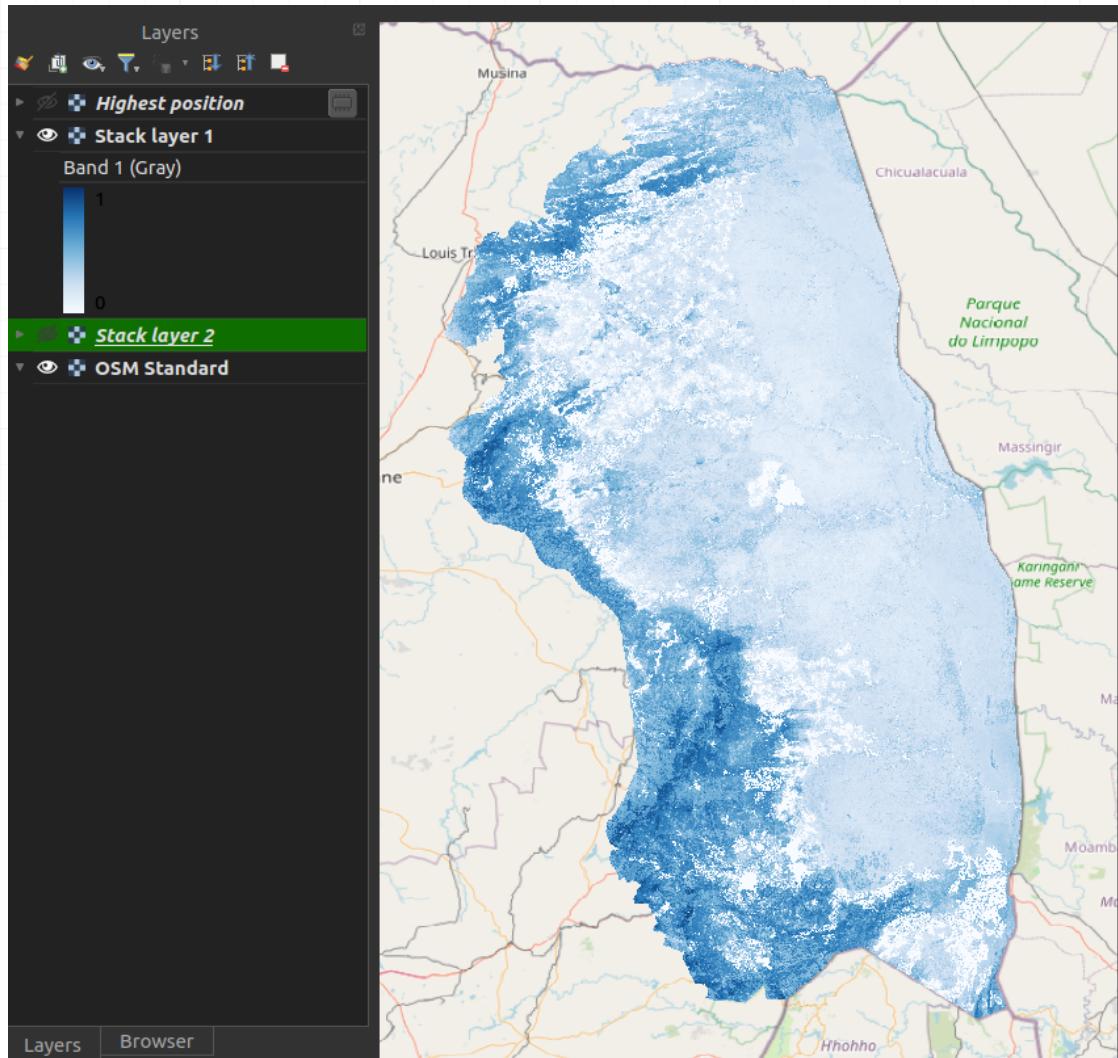


Figure 3: Layer 1 used as highest position input

- Figure 4 shows the layer for the Highest position at stack position 2

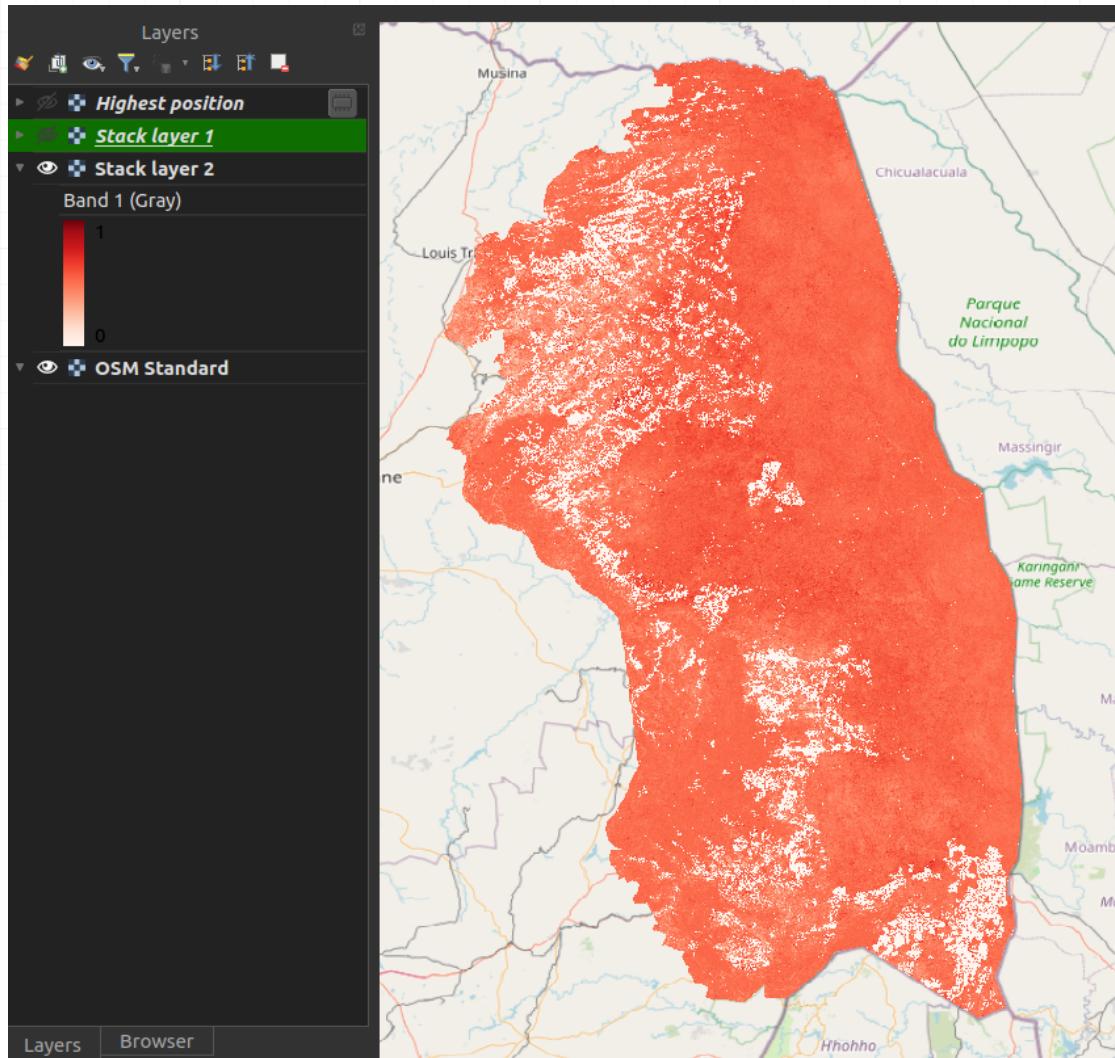


Figure 4: Layer 2 used as highest position input

- Figure 5 shows the result from the Highest position calculation (Scenario result)
 - Stack layer 1 (blue): Figure 2 raster had the highest pixel value
 - Stack layer 2 (red): Figure 3 raster had the highest pixel value

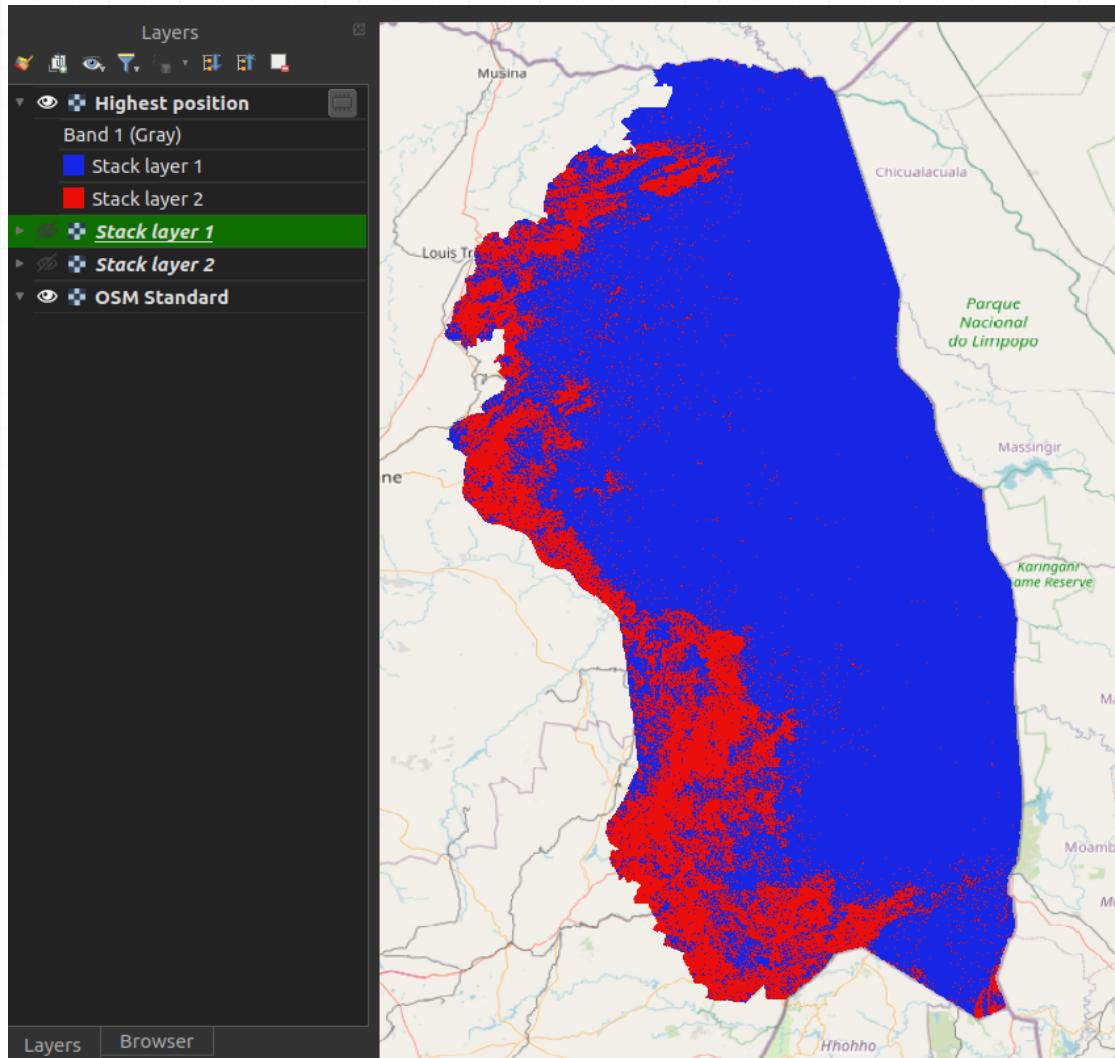


Figure 5: Highest position result

This concludes the section on how the calculations is done

References

- <https://www.pnas.org/doi/10.1073/pnas.1710465114>
- <https://royalsocietypublishing.org/doi/10.1098/rstb.2019.0126>

1.4.2 Plugin

Detailed descriptions for each UI element of the plugin. This covers steps 1 to 3, dialogs, and the settings UI.

Dock widget

This is the main UI of the plugin. The dock widget opens on the right side of QGIS. The dock widget consist of three tabs, each focussing on a particular phase of the analysis. Here is a short description of those steps:

- **Step 1:** Scenario information
- **Step 2:** NCS pathways and IMs
- **Step 3:** Weighting priorities (weighted IMs)

Step 1: Scenario information

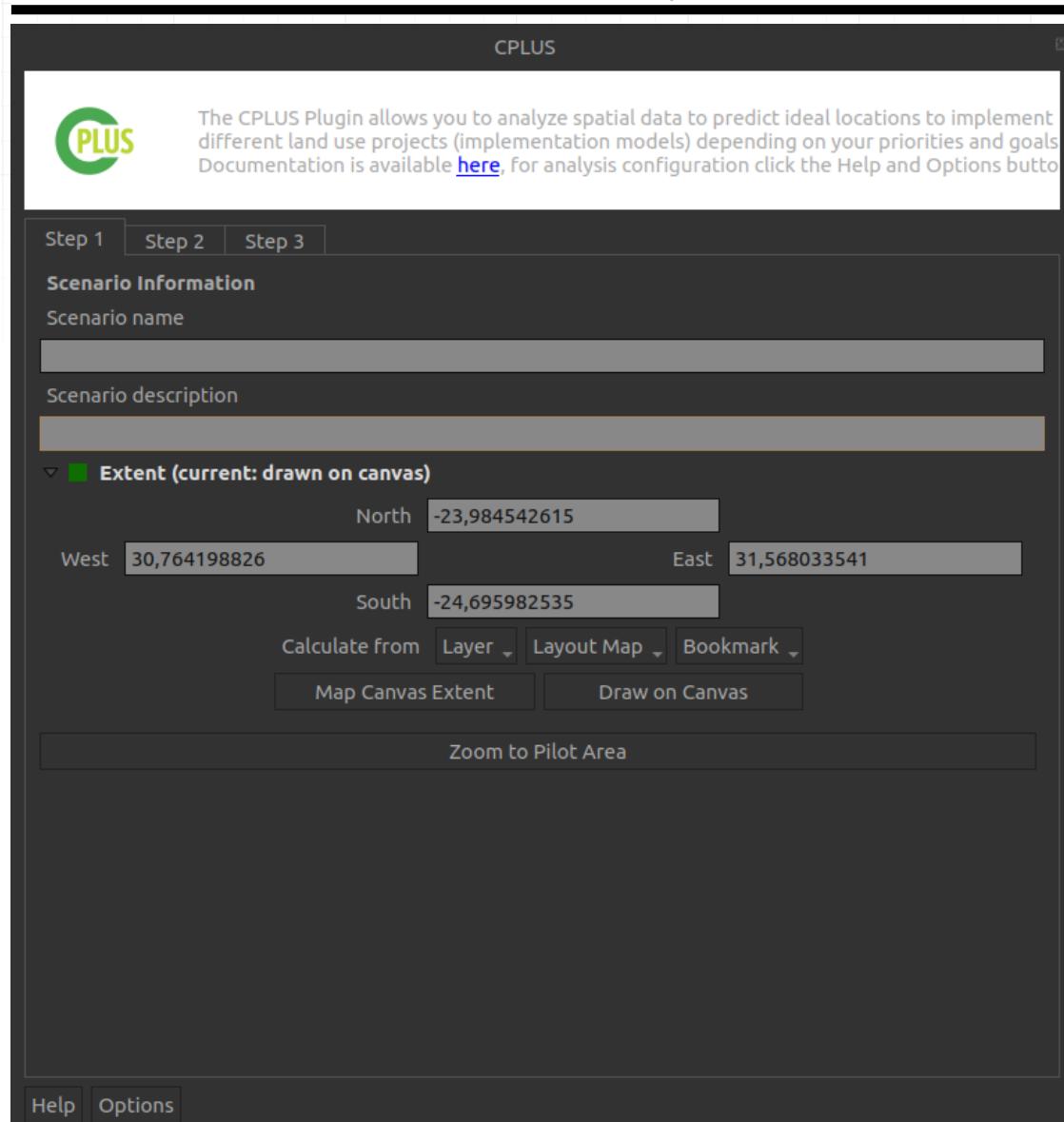


Figure 6: Step 1 of the dock widget

Step 1 allows a user to set up the scenario details and parameters.

- **Scenario name:** This title will be used throughout the processing, will be used for the groups added to the QGIS canvas, and in the generated report
- **Scenario description:** A detailed description of the scenario the user will be running. This information will be added to the final report
- **Extent:** The area of interest (AOI) for analysis. Any spatial data outside this region will be ignored
- **Map Canvas Extent:** The AOI will be the current extent the user has in QGIS
- **Draw on Canvas:** Allows the user to manually draw the AOI
- **Zoom to Pilot Area:** Zooms to the Bushbuckridge pilot study area

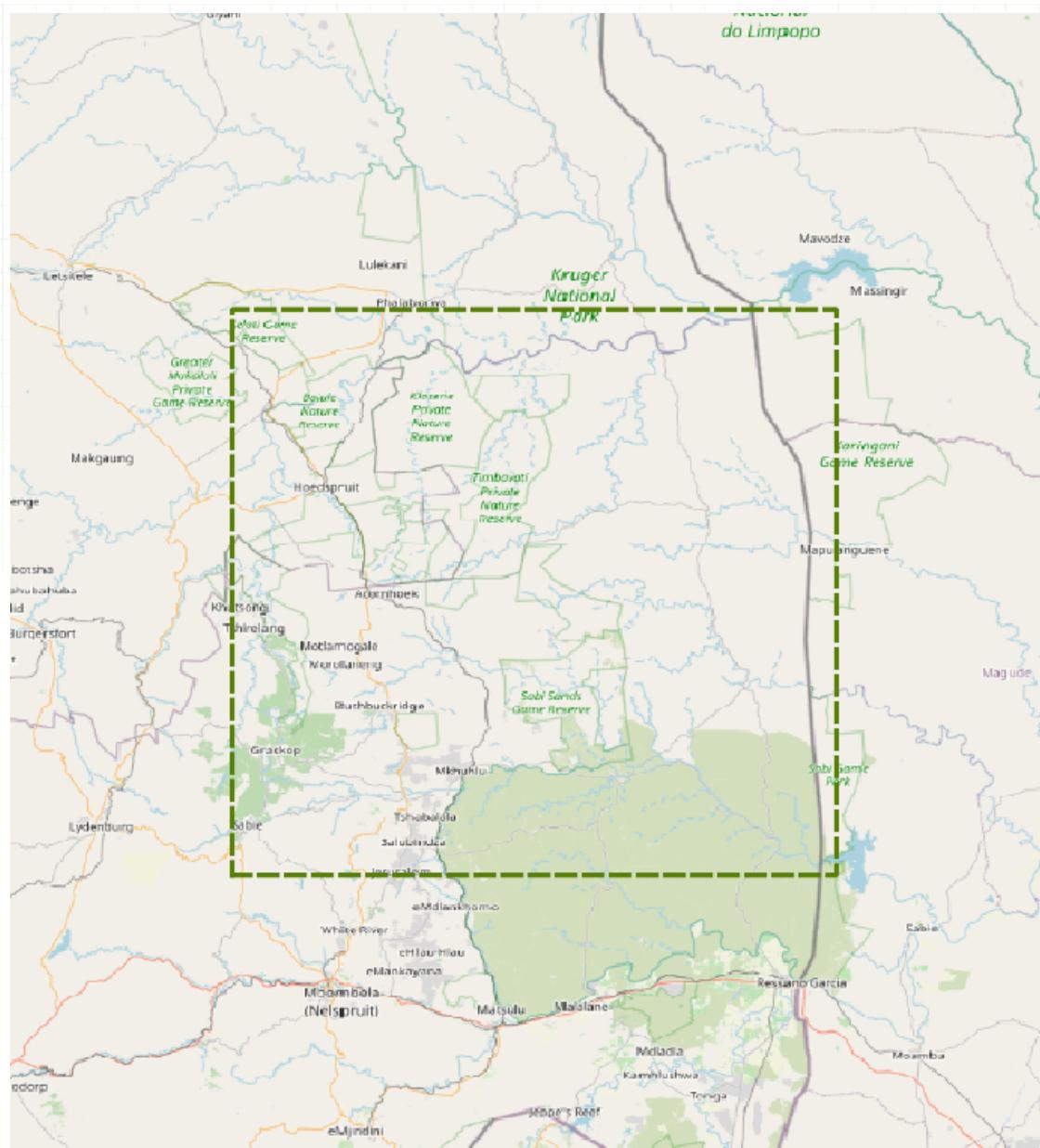


Figure 7: Bushbuckridge study area

Step 2: NCS pathways and Implementation models

Step 2 focuses on the implementation models (IMs) and pathways.

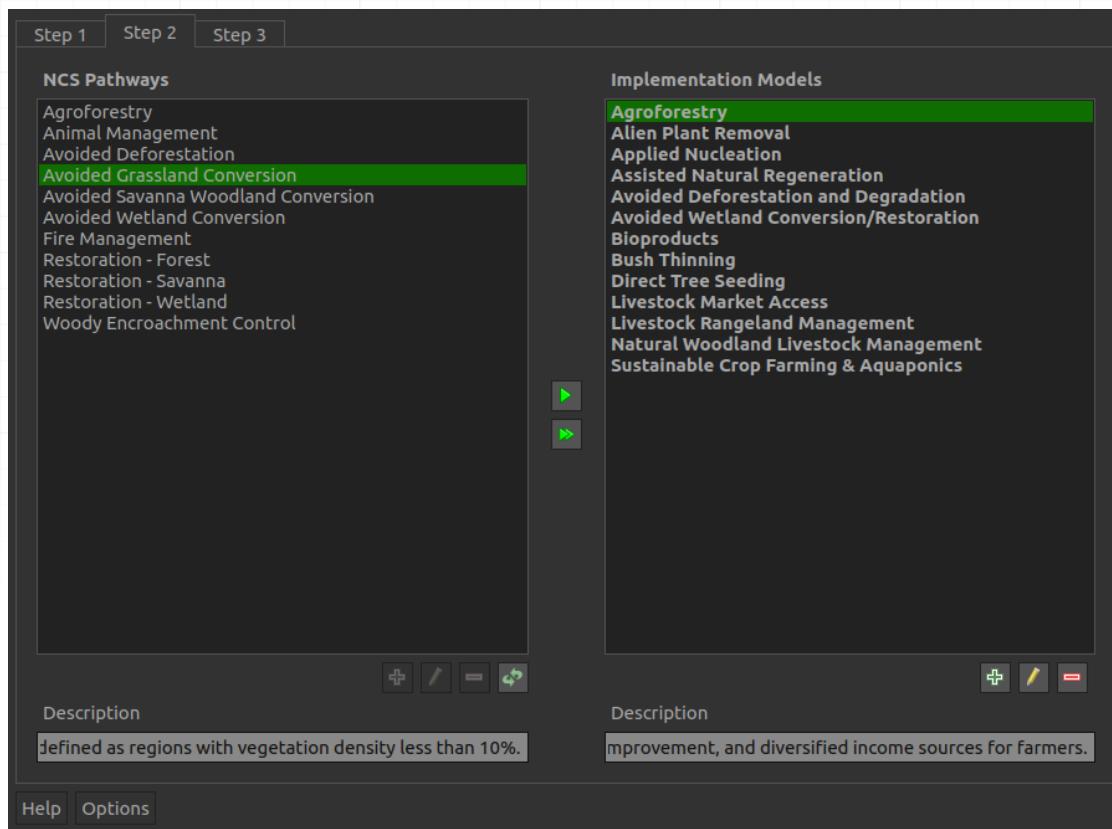


Figure 8: Step 2 of the dock widget

- **NCS pathways:** Pathways which can be added to IMs. Multiple pathways can be added to each IM
- **Implementation models:** Each selected model will be created in used to perform the analysis
- **Description:** A description of the IM or pathway selected
- : Add the selected pathway to the selected IM
- : Adds all pathways to the selected IM
- : Add a new IM
- : Remove the selected IM or pathway
- : Edit the selected IM
- : Order the pixel values (IMs) will be in the scenario output

Implementation Model Editor dialog

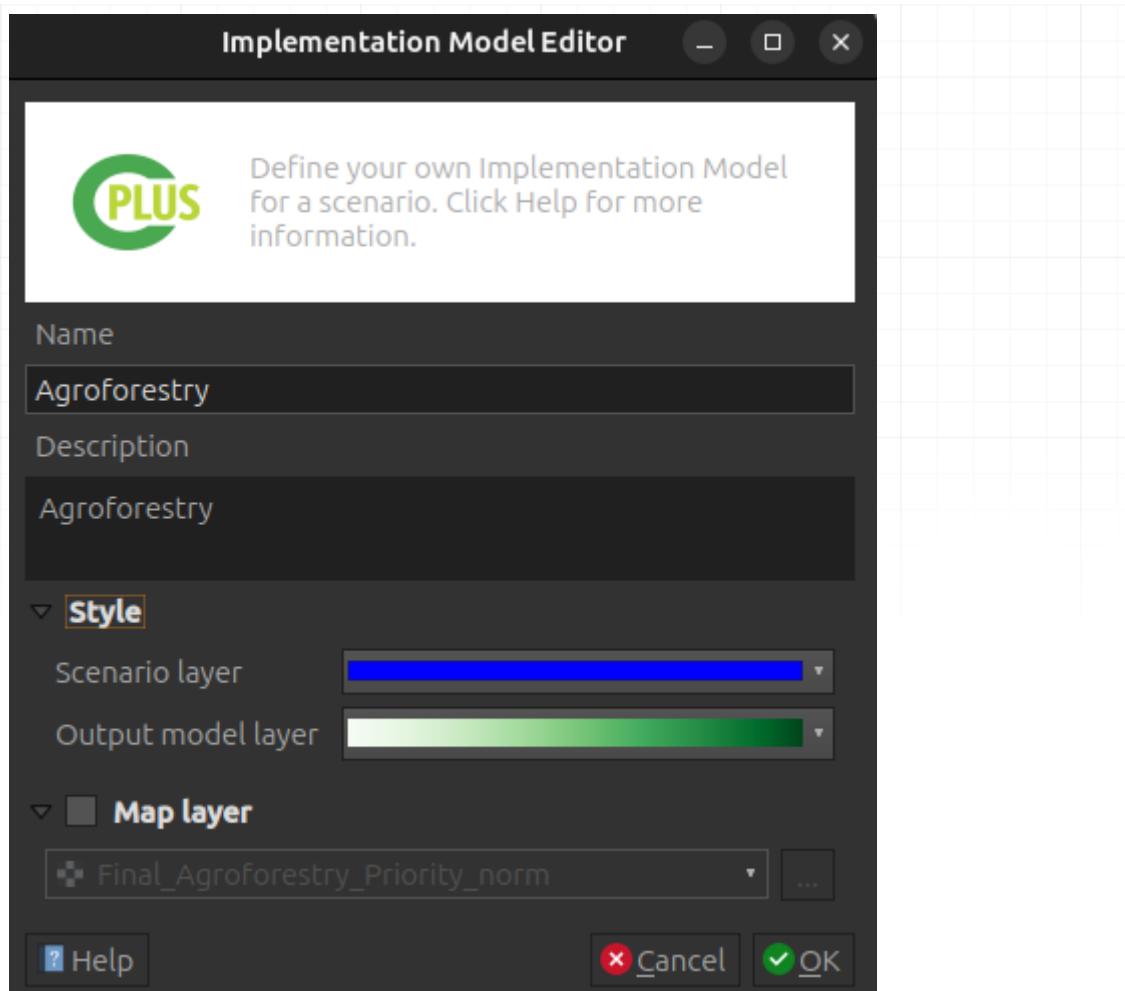


Figure 9: Implementation model editing/adding dialog

- **Name:** The name of the new IM or IM being edited. IM title will be used in the report
- **Description:** A detailed description of the IM. This will be used in the report
- **Style:** Styles used for the IM
 - Scenario layer: Colouring which will be used in the Scenario output for this IM
 - Output model layer: Colour ramp which will be applied to the IM raster output
- **Map layer:** If enabled, a user can provide an existing IM. This has to be a raster

Ordering of the pixel values for the scenario output

A user can order the stack using the **Style pixel value editor**.

Style Pixel Value Editor

Set the pixel value that will be used for styling the implementation models in the scenario layer by dragging and dropping the rows to specify the position.

1	Agroforestry
2	Alien Plant Removal
3	Applied Nucleation
4	Assisted Natural Regeneration
5	Avoided Deforestation and Degradation
6	Avoided Wetland Conversion/Restoration
7	Bioproducts
8	Bush Thinning
9	Direct Tree Seeding
10	Livestock Rangeland Management
11	Livestock Market Access
12	Natural Woodland Livestock Management
13	Protected Area Expansion
14	IM agroforestry
15	Sustainable Crop Farming & Aquaponics
16	IM open woodland
17	forest

? Help

Step 3: Weighting priorities

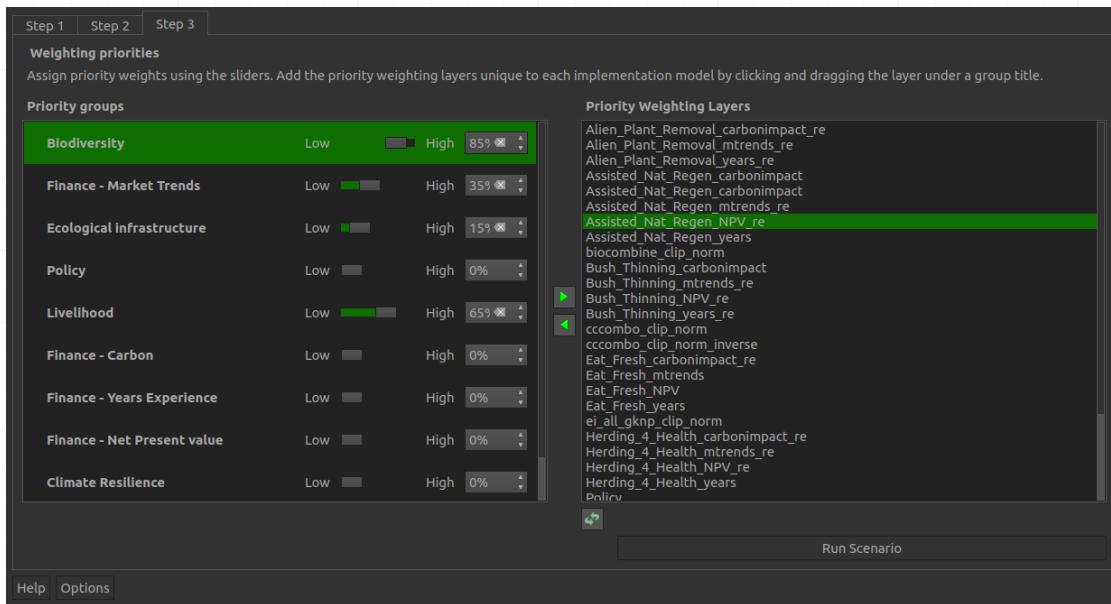


Figure 10: Step 3 of the dock widget

- **Priority groups:** Groups to which PWLs can be assigned
- **Priority weighted layers (PWL):** Importance of each priority group
- ➤ : Remove the selected PWL from the priority group
- ← : Add the selected PWL to the selected priority group
- + : Add a new PWL
- - : Remove the selected PWL
- 🖌 : Edit the selected PWL
- **Run Scenario:** Starts running the analysis. The progress dialog will open when the user clicks this button

Priority Weighted Layers Editor dialog

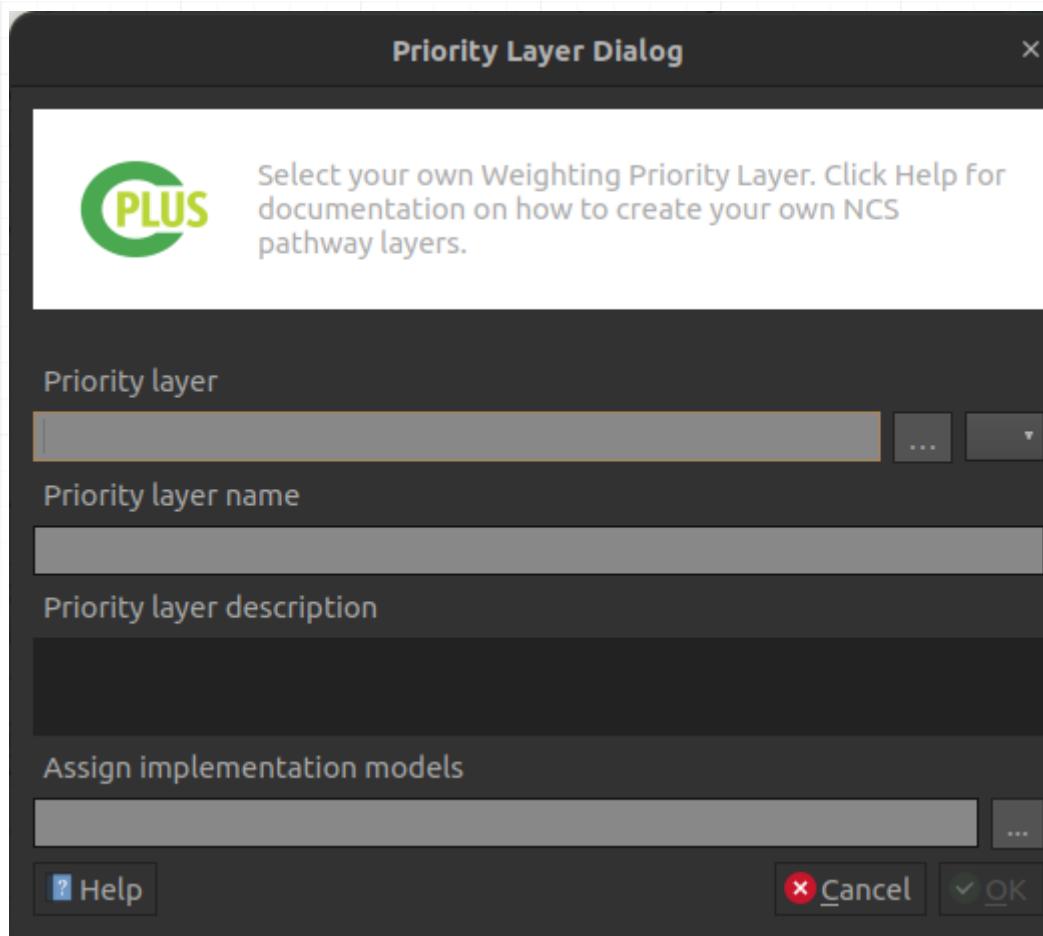


Figure 11: Priority layer dialog

- **Priority layer:** Select the priority layer
- **Priority layer name:** A unique name for the priority layer
- **Priority layer description:** A detailed description for the priority layer
- **Assign implementation models:** Selected IMs associated with the priority layer

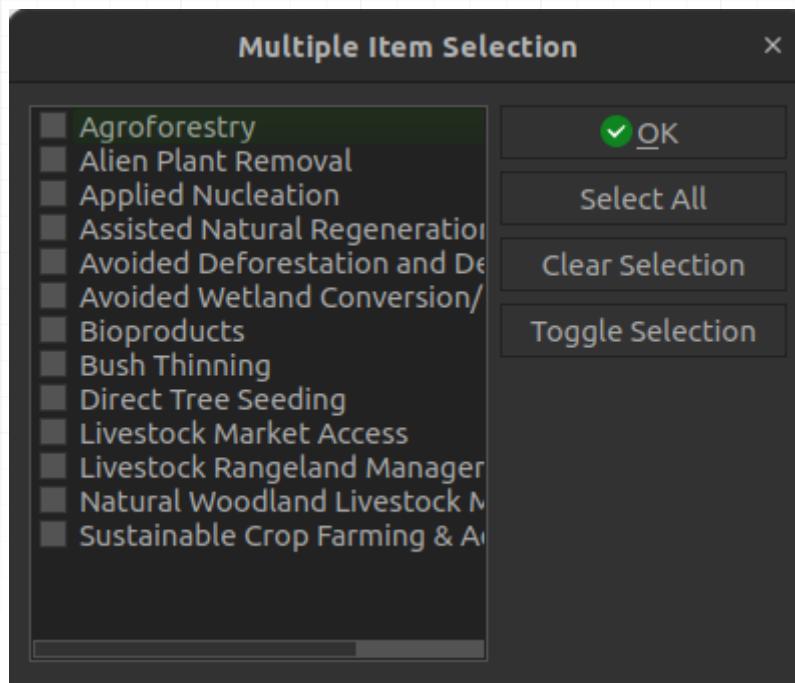


Figure 12: Selection of IMs for a custom priority layer

- List of IMs a user can select. Multiple IMs can be selected
- **OK:** Save the selected models
- **Select All:** Selects each of the available IMs
- **Clear Selection:** Deselects each of the selected IMs
- **Toggle Selection:** Switches each option from deselected to selected, or selected to deselected

Progress dialog

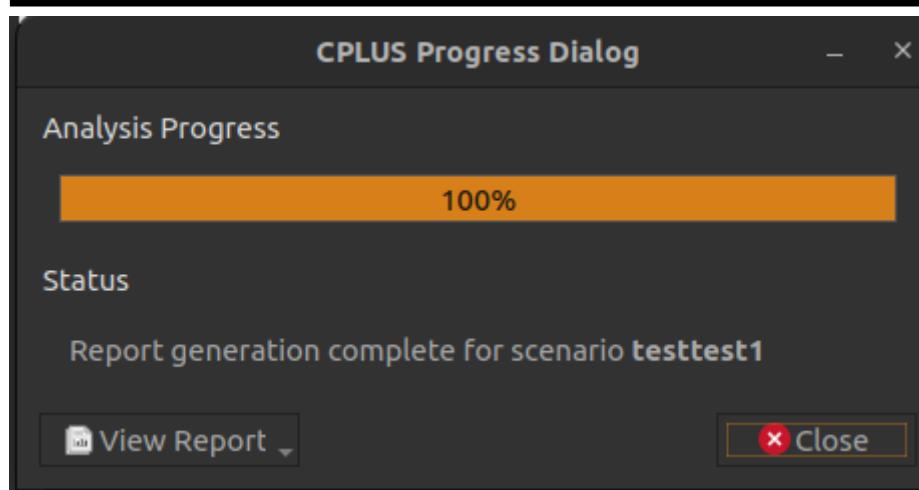


Figure 13: Processing dialog which will show the status of the analysis

- **Analysis Progress:** Progress of the current step
- **Status:** A status message on the current analysis being performed
- **View Report:** This button will remain disabled until the processing is done
- **Cancel:** Clicking this button will stop the processing
- **Close:** Only visible once the processing stops. Will close the progress dialog

Report options

These options will be available once the analysis has finished. The options will stay disabled if the analysis failed

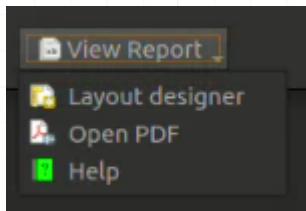


Figure 14: Options available to the user related to the generated report

- **Layout designer:** Opens the report in the QGIS layout designer
- **Open PDF:** Opens the created PDF
- **Help:** Takes the user to the Users documentation site

Settings

Reports

Organization: []

Contact email: []

Website: []

Custom logo: /home/divan/.local/share/QGIS/QGIS3/profiles/default/python/plugins/cplus_plugin/logos/ci_logo.png

Logo preview: 

Footer: []

Disclaimer: The boundaries, names, and designations used in this report do not imply official endorsement or acceptance by Conservation International Foundation, or its partner organizations and contributors.

License: Creative Commons Attribution 4.0 International License (CC BY 4.0)

Advanced

Base data directory: /home/divan/Kartoza/Projects/CI_CPLUS/base_folder2

Coefficient for carbon layers: 0.0

Pathway suitability index: 0.0

Snapping

Resample method: Nearest Neighbour

Reference layer: /an/Kartoza/Projects/CI_CPLUS/base_folder2/ncs_carbon/bou_SOC_carbonsum_norm_inverse_null_con_clip.tif

Rescale values

Figure 15: Settings available to the user

- **Reports:** Information to be added to the report
 - Organization: (optional) Organization or institute name
 - Contact email: (optional) Contact email of the user
 - Website: (optional) Link to website of your company or institute
 - Custom logo: (optional) If enabled, the user needs to provide a custom logo. Most formats should suffice (png, jpeg, etc.)
 - Logo preview: Visual previre of the default CI logo, or the custom logo a user selected
 - Footer: (optional) Will be added to the report
 - Disclaimer: Change as desired, otherwise use the default disclaimer
 - License: Change as desired, otherwise use the default license description
- **Advanced:**
 - Base data directory: Directory to read data from, and to which results will be written
 - Coefficient for carbon layers: Applied to carbon layers during processing
 - Pathway suitability index: Index multiplied to the pathways. Lower values means the pathway is less important, higher means its more important
 - Snapping: Will set rasters to match the cell alignment of a reference layer
 - Resample method: Resampling performed on pixel values.
 - Nearest neighbour: Closest pixel value. This will be best to use if a user wants to preserve the original pixel values
 - Bilinear: Computes the pixel values from the two closest pixels (2 x 2 kernel)
 - Cubic: Computes the pixel values from the four closest pixels (4 x 4 kernel)
 - Cubic B-Spline: Cubic resampling based on B-Spline (4 x 4 kernel)
 - Lanczos: Lanczos windowed sinc interpolation (6 x 6 kernel)
 - Average: Computes the average of all non-nodata contributing pixels
 - Mode: Selects the value which appears most often of all the sampled pixels
 - Maximum: Selects the maximum value which appears of all the sampled pixels
 - Minimum: Selects the minimum value which appears of all the sampled pixels
 - Mediane: Selects the mediane value which appears of all the sampled pixels
 - First quartile (Q1): Selects the first quartile value which appears of all the sampled pixels
 - Third quartile (Q3): Selects the third quartile value which appears of all the sampled pixels
 - Reference layer: The reference layer to which the cell alignment will be applied
 - Rescale values: Rescale values according to cell size

1 Administrators

1.1 Administrators

This section is for administrators of the plugin:

- The [guide](#) details how to create new tasks, and how to download and install a staging version of the plugin.
- The [repository](#) provides access to different versions of the plugin. This includes version based on specific pull requests.

1.2 Administrators guide

1.2.1 Pilot area data

This section deals with making updates to the data of the pilot study area (Bushback Ridge). This needs to be done in the GitHub repository, or locally on the repository clone, and then pushed into the repository. The change can be made as follows:

The following JSON files needs to be considered when doing this, all of which are stored in "src/cplus_plugin/data/default"

- implementation_model.json
- ncs_pathways.json
- priority_weighted_layers.json

Implementation models

In this file existing implementation models can be edited (e.g. change name or description), be removed or a new model can be added. Here is a quick overview of a model stored in the JSON file

- Each model contains the following elements:
 - **uuid**: A Universally unique identifier (UUID) for the model
 - **name**: A unique name for the model
 - **description**: Detailed description of the model
 - **pwls_ids**: UUIDs of the priority weighted layers associated with the IM
 - **style**: The style which will be applied to the
- Editing these will have an effect on the IM in the plugin

```
"models": [
  {
    "uuid": "a0b8fd2d-1259-4141-9ad6-d4369cf0dfd4",
    "name": "Agroforestry",
    "description": " Agroforestry is an integrated land use system th",
    "pwls_ids": [
      "c931282f-db2d-4644-9786-6720b3ab206a",
      "fce41934-5196-45d5-80bd-96423ff0e74e",
      "88c1c7dd-c5d1-420c-a71c-a5c595c1c5be",
      "9ab8c67a-5642-4a09-a777-bd94acf94acfae9d1",
      "2f76304a-bb73-442c-9c02-ff9945389a20"
    ],
    "style": {
      "scenario_layer": {
        "color": "#d80007",
        "style": "solid",
        "outline_width": "0",
        "outline_color": "35,35,35,0"
      },
      "model_layer": {"color_ramp": "Reds"}
    }
  },
]
```

When adding a new IM to the list (or a UUID needs to change), the user needs to provide a UUID. This can be done as follows:

- Open a [UUID generator](#). Other UUID generators can also be used, but the provided link will suffice
- Best will be to make use of version 4
- Click **Generate a version 4 UUID**
- Copy and paste the newly generated UUID

Version 4 UUID Generator

Generate a version 4 UUID

Bulk Version 4 UUID Generation

How Many? Generate [Download to a file](#)

bad38a5e-ac99-4bb1-a392-e922e29552a0

What is a version 4 UUID?

A Version 4 UUID is a universally unique identifier that is generated using random numbers. The Version 4 UUIDs produced by this site were generated using a secure random number generator.

To remove an IM from the list of models, an administrator can simply remove the entry in the JSON file. Remove this text to remove a model:

```
{  
    "uuid": "a0b8fd2d-1259-4141-9ad6-d4369cf0dfd4",  
    "name": "Agroforestry",  
    "description": " Agroforestry is an integrated land use system tha  
    "pwls_ids": [  
        "c931282f-db2d-4644-9786-6720b3ab206a",  
        "fce41934-5196-45d5-80bd-96423ff0e74e",  
        "88c1c7dd-c5d1-420c-a71c-a5c595c1c5be",  
        "9ab8c67a-5642-4a09-a777-bd94acf9ae9d1",  
        "2f76304a-bb73-442c-9c02-ff9945389a20"  
    ],  
    "style": {  
        "scenario_layer": {  
            "color": "#d80007",  
            "style": "solid",  
            "outline_width": "0",  
            "outline_color": "35,35,35,0"  
        },  
        "model_layer": {"color_ramp": "Reds"}  
    }  
},
```

To update the list of priority weighted layers for a model, the ID needs to be retrieved:

- Open the priority_weighted_layers.json file
- Each available PWL will be listed under "layers"
- Copy and paste the UUID that needs to be added to the IM pwls_ids field
- Save the file

```
{  
  "layers": [  
    {  
      "uuid": "c931282f-db2d-4644-9786-6720b3ab206a",  
      "name": "social_int_clip_norm",  
      "description": "Placeholder text for social_int_clip_norm",  
      "selected": true,  
      "path": "social_int_clip_norm.tif"  
    },  
    {  
      "uuid": "f5687ced-af18-4cf8-9bc3-8006e40420b6",  
      "name": "social_int_clip_norm_inverse",  
      "description": "Placeholder text for social_int_clip_norm_inverse",  
      "selected": false,  
      "path": "social_int_clip_norm_inverse.tif"  
    },  
    {  
      "uuid": "fef3c7e4-0cdf-477f-823b-a99da42f931e",  
      "name": "cccombo_clip_norm_inverse",  
      "description": "Placeholder text for cccombo_clip_norm_inverse",  
      "selected": false,  
      "path": "cccombo_clip_norm_inverse.tif"  
    }  
  ]  
}
```

A section on PWL editing will soon follow.

NCS pathways

An administrator can access/edit the NCS pathways as follows:

- Open the ncs_pathways.json file
- Here is a description of each element:
 - **uuid**: A unique identifier for the pathway
 - **name**: Unique title for the pathway
 - **description**: Detailed description of the pathway
 - **path**: Directory with file name for the pathway data
 - **layer_type**: Zero (0) for rasters, one (1) for vector layers
 - **carbon_paths**: A list of the carbon footprint rasters. This should be a directory with the raster name. Tif is the preferred format
- Editing these elements will make changes to the pilot area data in the plugin

```
"pathways": [
  {
    "uuid": "b187f92f-b85b-45c4-9179-447f7ea114e3",
    "name": "Agroforestry",
    "description": "Provides additional carbon sequestration in agricultural systems by strategically planting trees in croplands.",
    "path": "Final_Agroforestry_Priority_norm.tif",
    "layer_type": 0,
    "carbon_paths": ["bou_SOC_carbonsum_norm_inverse_null_con_clip.tif"]
  }
],
```

When adding a new pathway to the list (or a UUID needs to change), the user needs to provide a UUID. This can be done as follows:

- Open a [UUID generator](#). Other UUID generators can also be used, but the provided link will suffice
- Best will be to make use of version 4
- Click **Generate a version 4 UUID**
- Copy and paste the newly generated UUID

The screenshot shows a dark-themed web application for generating UUIDs. At the top, it says "Version 4 UUID Generator". Below that is a button labeled "Generate a version 4 UUID". Underneath, there's a section titled "Bulk Version 4 UUID Generation" with a "How Many?" input field set to "Max 500", a "Generate" button, and a "Download to a file" link. A generated UUID, "bad38a5e-ac99-4bb1-a392-e922e29552a0", is displayed. At the bottom, there's a question "What is a version 4 UUID?" followed by a detailed explanation: "A Version 4 UUID is a universally unique identifier that is generated using random numbers. The Version 4 UUIDs produced by this site were generated using a secure random number generator."

To remove a pathway from the list of layers, an administrator can simply remove the entry in the JSON file. Remove this text to remove a model:

```
{
    "uuid": "b187f92f-b85b-45c4-9179-447f7ea114e3",
    "name": "Agroforestry",
    "description": "Provides additional carbon sequestration in agricultural systems by strategically planting trees in croplands.",
    "path": "Final_Agroforestry_Priority_norm.tif",
    "layer_type": 0,
    "carbon_paths": ["bou_SOC_carbonsum_norm_inverse_null_con_clip.tif"]
},
```

Priority weighted layers

Do the following to change/add/remove priority weighted layers for the pilot study area

- Open the priority_weighted_layers.json file
- Here is a description of each element:
 - **uuid**: A unique ID for the PWL
 - **name**: Unique name for the PWL
 - **description**: A detailed description of the PWL
 - **selected**: Whether the PWL should be selected on default
 - **path**: A directory with file name for the PWL data. This should be a raster
- Changing the above values will have an impact on the data in the plugin for the pilot study area

```
"layers" : [
{
    "uuid": "c931282f-db2d-4644-9786-6720b3ab206a",
    "name": "social_int_clip_norm",
    "description": "Placeholder text for social_int_clip_norm",
    "selected": true,
    "path": "social_int_clip_norm.tif"
},
```

When adding a new PWL to the list (or a UUID needs to change), the user needs to provide a UUID. This can be done as follows:

- Open a [UUID generator](#). Other UUID generators can also be used, but the provided link will suffice
- Best will be to make use of version 4
- Click **Generate a version 4 UUID**
- Copy and paste the newly generated UUID

To remove a PWL from the list of layers, an administrator can simply remove the entry in the JSON file.
Remove this text to remove a model:

```
{
  "uuid": "c931282f-db2d-4644-9786-6720b3ab206a",
  "name": "social_int_clip_norm",
  "description": "Placeholder text for social_int_clip_norm",
  "selected": true,
  "path": "social_int_clip_norm.tif"
},
```

1.2.2 Bugs and suggestions

This section relates to creating an issue for when a bug is found in the plugin, or if the user has a suggested improvement for the plugin.

- Go to the [CPLUS repository](#)
- Click on the **Issues** tab
- Click on **New Issue** (see **Figure 1**)
- Title: Short, but descriptive
- Description: Detailed description. If it's a bug, an explanation on how to replicate the bug will be best. Screenshots of the bug or suggestion will also be helpful

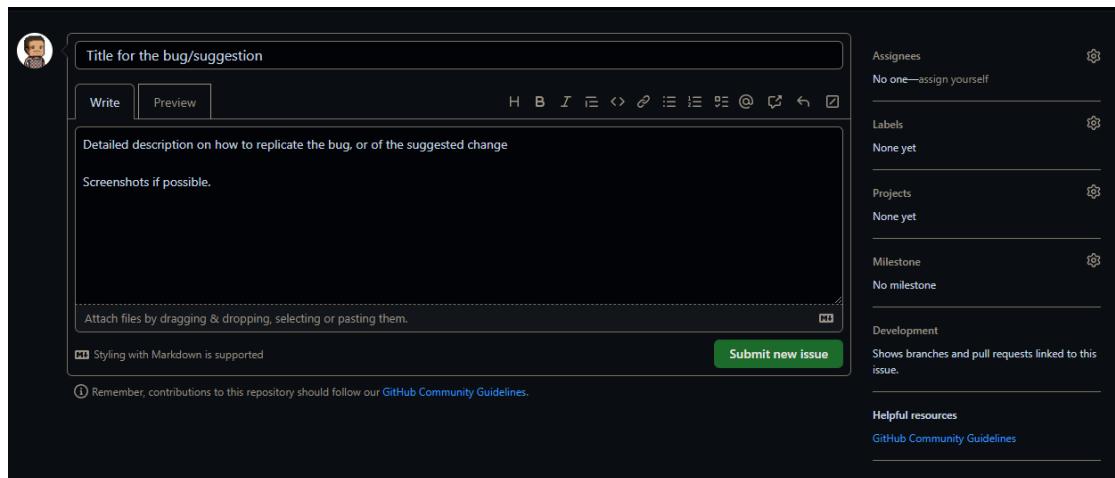


Figure 1: An example of a new GitHub issue

- Select a **Label** (e.g. bug, enhancement, etc.) as shown in **Figure 2**

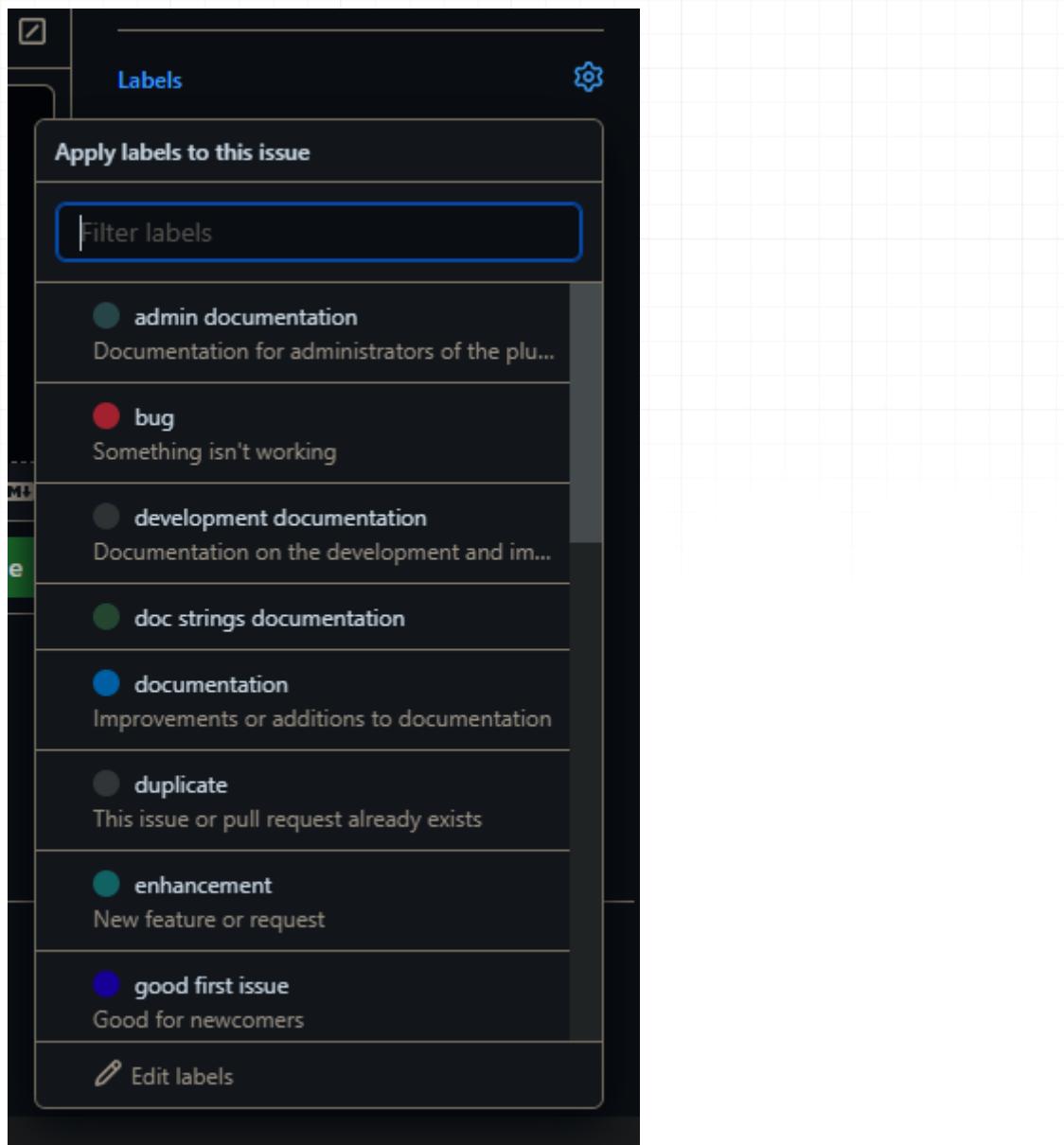


Figure 2: Selecting a label for an issue

- Select the CPLUS **Project** (**Figure 3**). This will add the issue/task to the project board

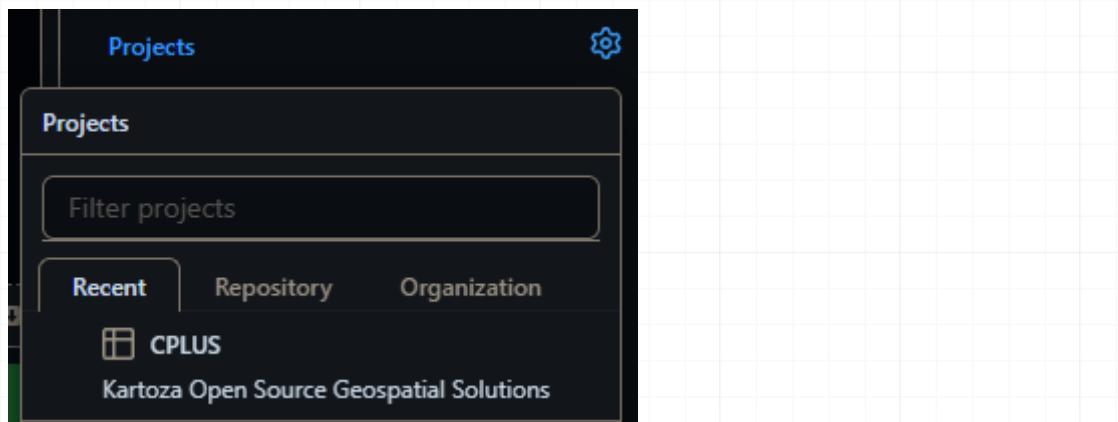


Figure 3: Selecting a Project for an issue

- The end result should be similar to **Figure 4**.

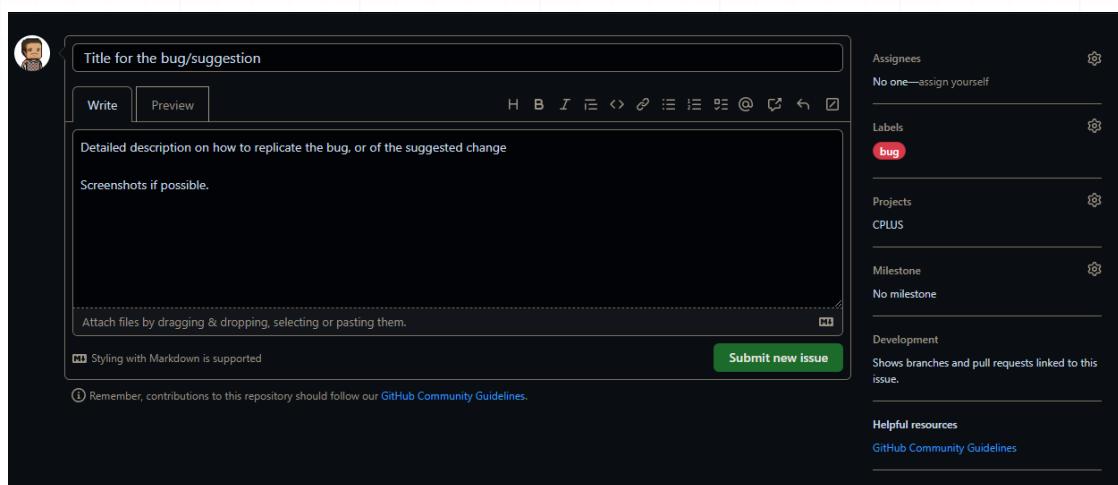


Figure 4: An example of a finalized issue

- Click **Submit new issue**

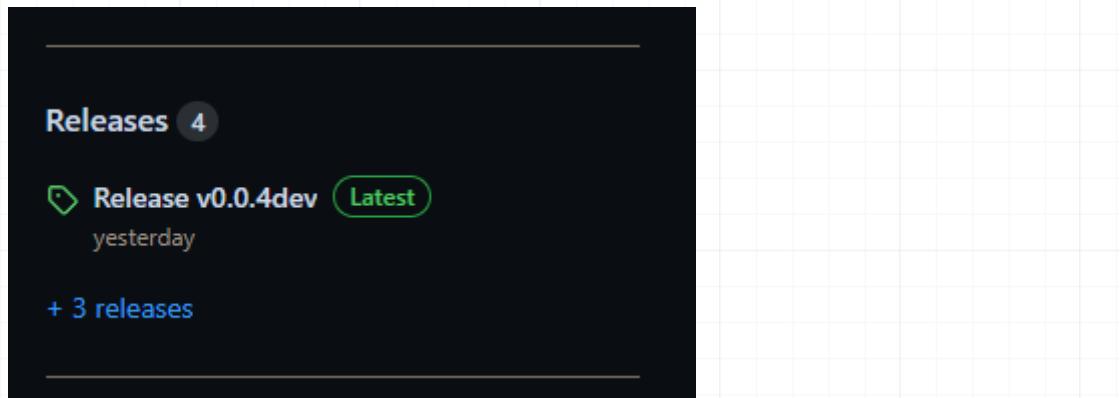
The issue will now be submitted to the GitHub repository and be available to the developers.

1.2.3 Staging version of the plugin

When a pull requested is performed, an automatic staging version is created. This will allow a developer to test their changes to the plugin with other changes which has not been merged into the main branch. Another advantage of this approach is to show the client to progress of the plugin.

Get the staging version

-
- Go to the repository: <https://github.com/kartoza/cplus-plugin>
 - To the right there is a section named **Releases**



- Click on **Latest** release
- Download the cplus_plugin.zip file if you want to install the plugin in QGIS
- Developers will likely be interested in Source code (zip) and Source code (tar.gz) options

Release v0.0.5dev Latest

github-actions released this yesterday v0.0.5dev -o- f4b8228

version 0.0.5 for development

Assets 3

	254 KB	yesterday
cplus_plugin.0.0.1.zip		
Source code (zip)		yesterday
Source code (tar.gz)		yesterday

- See user/installation on how to install a QGIS plugin

If you want to have a look at past versions of the plugin:

- On the repository page, click on **Releases**
- A list of option will appear
- Choose the version you are interested in, and follow the steps discussed above

The screenshot shows the GitHub Releases interface for a repository. At the top, there are tabs for 'Releases' (which is selected) and 'Tags'. On the right, there are buttons for 'Draft a new release' and 'Find a release'.

The first release listed is 'Release v0.0.2dev' (Draft), created '2 weeks ago' by 'github-actions'. It includes a commit log entry: 'v0.0.2dev' with hash '21963d3'. A 'Compare' button is available.

The second release listed is 'Release v0.0.5dev' (Latest), created 'yesterday' by 'github-actions'. It includes a commit log entry: 'v0.0.5dev' with hash 'f4b8228'. A 'Compare' button is available.

Both releases have an 'Assets' section. The 'v0.0.5dev' release has three assets:

Asset	Size	Last Updated
cplus_plugin.0.0.1.zip	254 KB	yesterday
Source code (zip)		yesterday
Source code (tar.gz)		yesterday

1.2.1 Pull requests artifacts

PR title **PR url** **Artifact name** **Artifact link** **Created date**

1.2.1 Main branch artifacts

Commit link **Artifact name** **Artifact link** **Created date**

1 Developers

1.1 Developers

Section which aims at guiding developers of the plugin:

- [Setup](#) details how to set up a developers environment of the plugin.
- [Developers documentation](#) details how a developer should document changes to the code.
- [Architecture of the plugin](#)

1.2 Setup

build no status license [GPL-3.0](#)

To use the plugin for development purposes, clone the repository locally, install pip, a python dependencies management tool see <https://pypi.org/project/pip/>

1.2.1 Create virtual environment

Using any python virtual environment manager create project environment. Recommending to use [virtualenv-wrapper](#).

It can be installed using python pip

 **Code:**

```
pip install virtualenvwrapper
```

1. Create virtual environment

```
mkvirtualenv cplus
```

2. Using the pip, install plugin development dependencies by running

```
pip install -r requirements-dev.txt
```

To install the plugin into the QGIS application, activate virtual environment and then use the below command

 **Code:**

```
python admin.py install
```

1.3 Architecture

1.3.1 Frameworks used

1.3.2 High-level system architecture

1.3.3 Data model

1.4 Working with documentation

Documentation is written using [mkdocs](#). A detailed description on getting-started with mkdocs is available [here](#). Developer documentation will be created and generated using [mkdocstrings](#).

1.4.1 Install mkdocs

- Open the terminal
- Run "pip install mkdocs"
- This should install mkdocs and all requirements

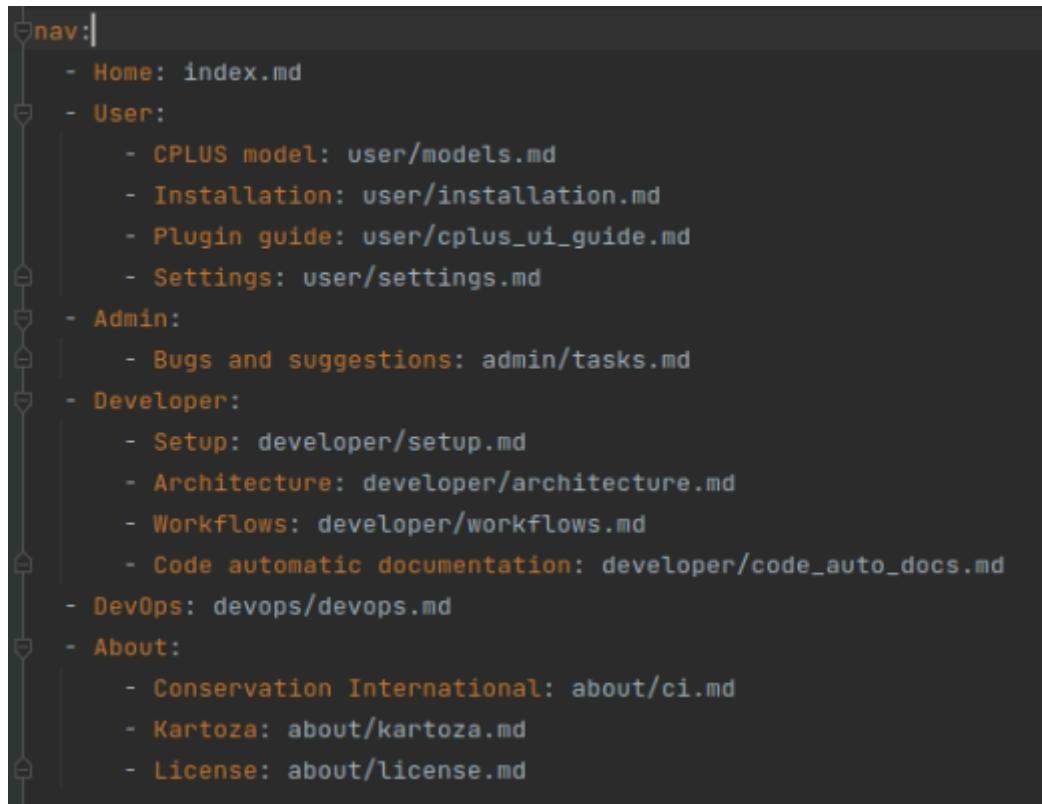
1.4.2 Creating a new project

This should not be required as the mkdocs has been created already, but serves more of a guide for a user whom are new to mkdocs.

- Open the terminal
- Run "mkdocs new ."
- This will generate the documents folder with the home page index markdown file

1.4.3 Updating the mkdocs.yml file

Mostly, the only changes a user will need to make is to the **nav** section in the mkdocs.yml file (**Figure 1**). Other options, such as the themes, plugins and extensions, should require no changes.



```
nav:
  - Home: index.md
  - User:
    - CPLUS model: user/models.md
    - Installation: user/installation.md
    - Plugin guide: user/cplus_ui_guide.md
    - Settings: user/settings.md
  - Admin:
    - Bugs and suggestions: admin/tasks.md
  - Developer:
    - Setup: developer/setup.md
    - Architecture: developer/architecture.md
    - Workflows: developer/workflows.md
    - Code automatic documentation: developer/code_auto_docs.md
  - DevOps: devops/devops.md
  - About:
    - Conservation International: about/ci.md
    - Kartoza: about/kartoza.md
    - License: about/license.md
```

Figure 1: Navigation example of mkdocs

Here is an explanation on how the **nav** should be updated:

- It is important to keep to the structure of the mkdocs
- Each section focusses on a particular aspect, for example **User** will provide information on installing the plugin, a guide on how to use the plugin, etc.
- The order in which the **nav** is structured will determine the tabs order on the site, as shown in **Figure 2**

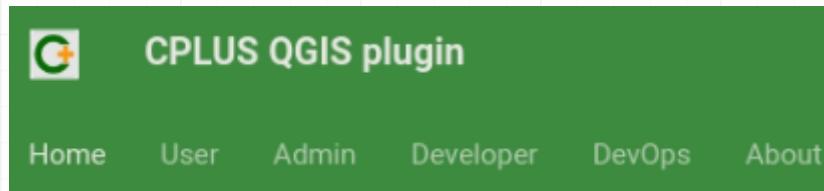


Figure 2: Tabs in a generated site

- Each tab, or group, will then be structured as in **Figure 3**

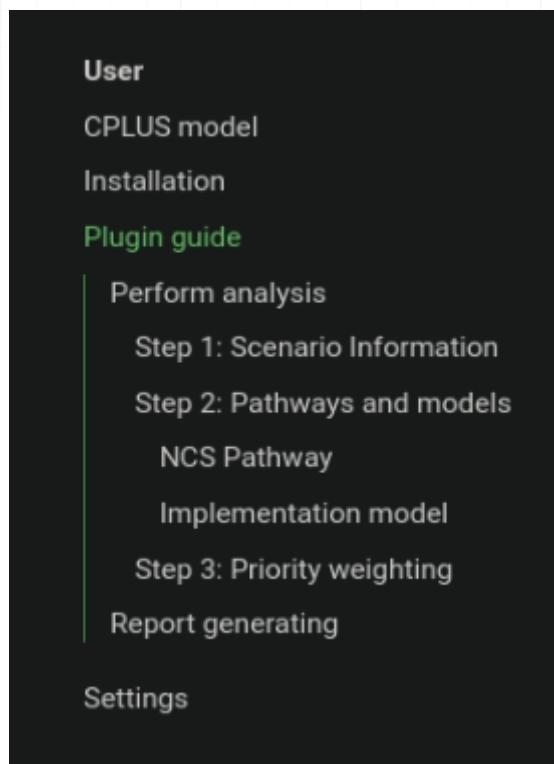


Figure 3: Structure created by markdown headings

- The additional titles (in the Plugin guide section in the above example) is based on the headings provided in the guide markdown file (cplus_ui_guide.md in this case)
- Headings for each section should be short, but descriptive
- The section itself can have a more detailed description
- Add screenshots where possible. This will make the guide(s) user-friendly

1.4.4 mkdocstrings

This is an important aspect of documenting code. Using mkdocstrings is an easy and effective way of keeping track of classes, functions, etc. Any changes to the code will automatically update here, but the developer needs to make the changes in the code (comments) for this to happen.

The comments for mkdocstrings is three sections:

- Description: A description on what the function does. A detailed description are welcome
- param: List of parameters for the function. Type and description should be included
- returns: A list of values which the function would return. Type should be included, with a description

```
▲ vermeulendivan
def createWidget(self, parent: QWidget) -> CplusSettings:
    """Creates a widget for CPLUS settings.

    :param parent: Parent widget
    :type parent: QWidget

    :returns: Widget to be used in the QGIS options
    :rtype: CplusSettings
    """

    return CplusSettings(parent)
```

Here is an example of the end-result:

The screenshot shows the mkdocstrings options interface. At the top, it displays the function signature `createWidget(parent)`. Below this, a detailed description states: "Creates a widget for CPLUS settings." Under the heading "Parameters:", there is a table:

Name	Type	Description	Default
parent	QWidget	Parent widget	<i>required</i>

Under the heading "Returns:", there is another table:

Type	Description
CplusSettings	Widget to be used in the QGIS options

At the bottom, a code editor window shows the source code for the `createWidget` function:

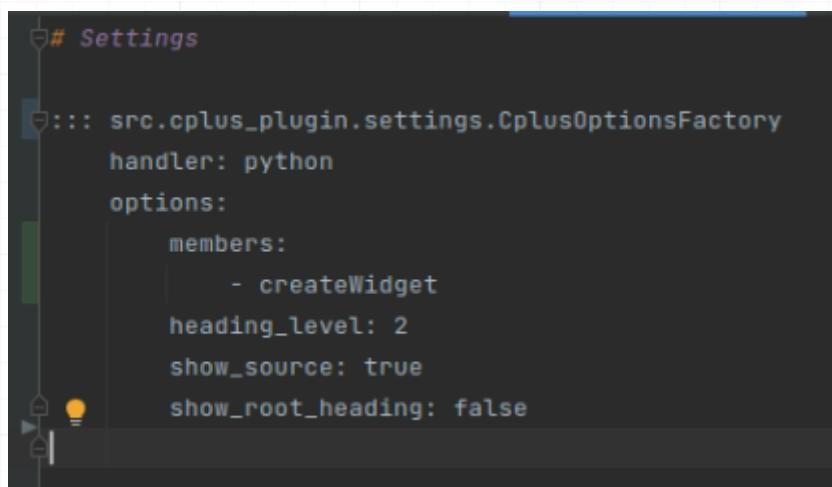
```
## Source code in src/cplus_plugin/settings.py

335 def createWidget(self, parent: QWidget) -> CplusSettings:
336     """Creates a widget for CPLUS settings.
337     :param parent: Parent widget
338     :type parent: QWidget
339
340     :returns: Widget to be used in the QGIS options
341     :rtype: CplusSettings
342     """
343
344     return CplusSettings(parent)
```

mkdocstrings options

The user can also do fine-tuning of the resulting site. Here is a list of important options to be aware of: `members`, `show_root_directory`, `show_source`, and `heading_level`.

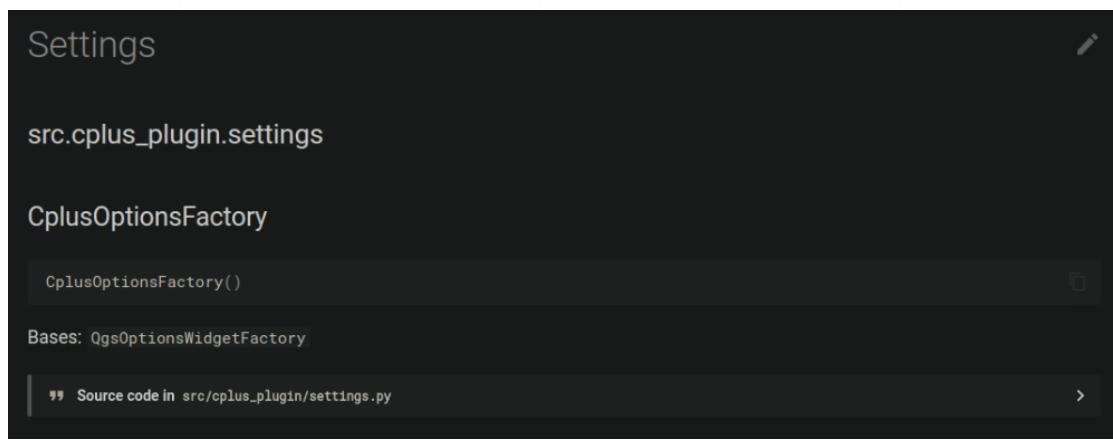
- **heading_level**: The header level of the content (functions, etc.) which will be added
- **members**: Only these function/classes will be shown



```
# Settings

::: src.cplus_plugin.settings.CplusOptionsFactory
    handler: python
    options:
        members:
            - createWidget
    heading_level: 2
    show_source: true
    show_root_heading: false
```

- **show_root_directory**: The root directory of the code
- Best will be to disable this, as it looks neater
- Here is an example of show_root_directory enabled, followed by an example when it is disabled:

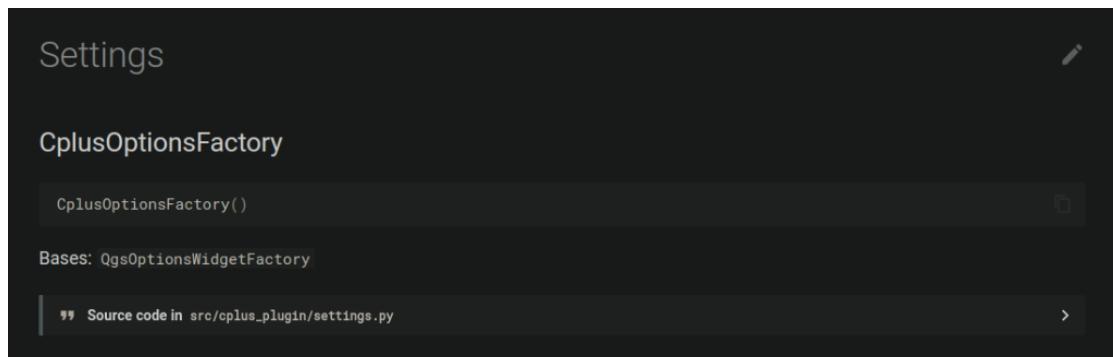


Settings

```
src.cplus_plugin.settings

CplusOptionsFactory

    CplusOptionsFactory()
    Bases: QgsOptionsWidgetFactory
    " Source code in src/cplus_plugin/settings.py >
```

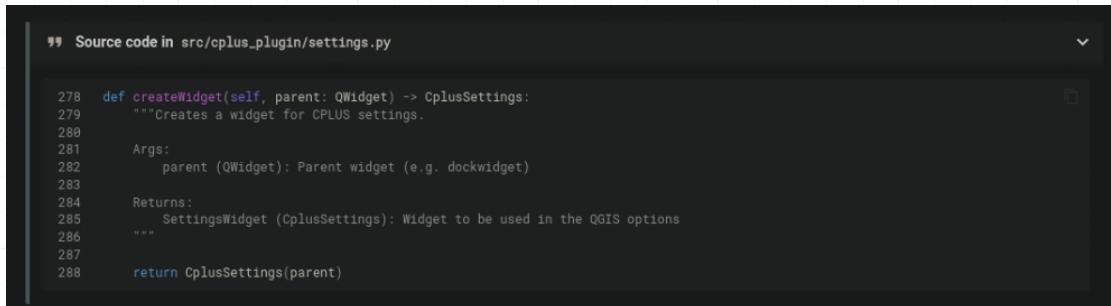


Settings

```
CplusOptionsFactory

    CplusOptionsFactory()
    Bases: QgsOptionsWidgetFactory
    " Source code in src/cplus_plugin/settings.py >
```

- **show_source**: Shows the directory of the source code and a snippet of the code



```
## Source code in src/cplus_plugin/settings.py

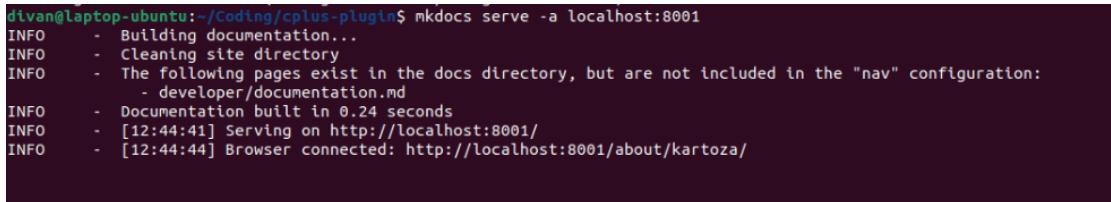
278 def createWidget(self, parent: QWidget) -> CplusSettings:
279     """Creates a widget for CPLUS settings.
280
281     Args:
282         parent (QWidget): Parent widget (e.g. dockwidget)
283
284     Returns:
285         SettingsWidget (CplusSettings): Widget to be used in the QGIS options
286
287     """
288     return CplusSettings(parent)
```

More information on this can be found [here](#).

1.4.5 Serving the pages locally

This step is useful when making changes and the user wants to test and review their changes to the mkdocs before creating a pull request.

- Open the terminal
- Run "mkdocs serve"
- **Figure 4** shows an example of the result



```
divan@laptop-ubuntu:~/Coding/cplus-plugin$ mkdocs serve -a localhost:8001
INFO    - Building documentation...
INFO    - Cleaning site directory
INFO    - The following pages exist in the docs directory, but are not included in the "nav" configuration:
        - developer/documentation.md
INFO    - Documentation built in 0.24 seconds
INFO    - [12:44:41] Serving on http://localhost:8001/
INFO    - [12:44:44] Browser connected: http://localhost:8001/about/kartoza/
```

Figure 4: Console example of serving mkdocs

- On default, mkdocs is served to localhost:8000. But if the port is already in use, the user needs to provide a port number
- Run "mkdocs serve -a localhost:8001". The user can use a port number of their choosing
- Open the URL in a browser: localhost:8000
- The Home page should be similar to **Figure 5**

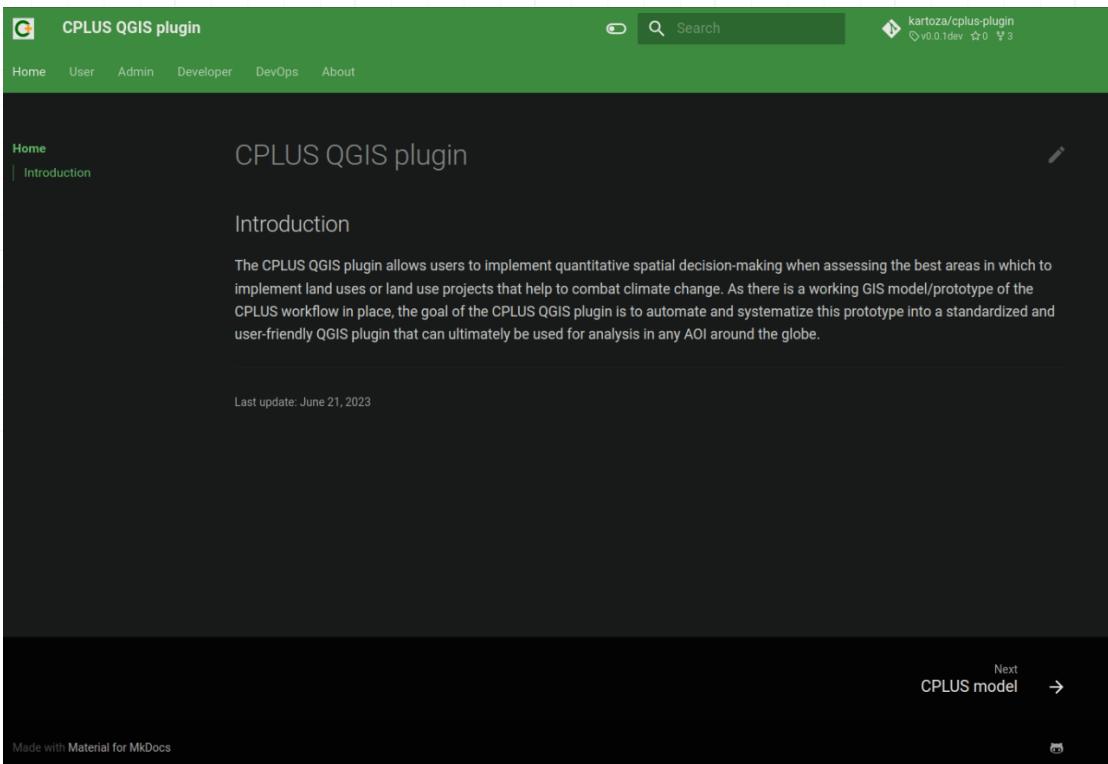


Figure 5: Site example. This is the Home page

Errors

When performing the serve, mkdocs will automatically check for any errors. An example will be when a file linked in the documentation cannot be found (**Figure 6**).

```
divan@laptop-ubuntu:~/coding/cplus.plugin$ mkdocs serve -a localhost:8001
INFO  - Building documentation...
INFO  - Cleaning site directory
INFO  - The following pages exist in the docs directory, but are not included in the "nav" configuration:
      - developer/documentation.md
WARNING - Documentation file 'developer/documentation.md' contains a link to 'img/documentation/mkdocs-pages-example222.png' which is not found in the documentation files.
INFO  - Documentation built in 0.24 seconds
INFO  - [12:54:24] Serving on http://localhost:8001/
INFO  - [12:54:27] Browser connected: http://localhost:8001/
```

Figure 6: Missing file example when serving mkdocs

Be sure to check for such errors in the console prior to creating a pull request for your documentation changes.

1.4.6 GitHub pages

This is only required if it has not been set up on GitHub for the repository, or if it has been disabled.
Only a user with admin rights to the repository will be able to do this.

- Go to the repository and click on **Settings**
- Click on **Pages**
- Set the branch to "gh-pages"
- Click **Save**
- Select the action
- Select **Deploy**
- Open the Run mkdocs gh-deploy section
- The URL should be <https://kartoza.github.io/cplus-plugin/>



1.5 API

1.5.1 Core Main

QGIS CPLUS Plugin Implementation.



- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/main.py`

Code:

```
88
89     def __init__(self, iface):
90         self.iface = iface
91         self.plugin_dir = os.path.dirname(__file__)
92         locale = QgsSettings().value("locale/userLocale")[0:2]
93         locale_path = os.path.join(self.plugin_dir, "i18n", "CPLUS{}")
94
95
96
97     if os.path.exists(locale_path):
98         self.translator = QTranslator()
99         self.translator.load(locale_path)
100        QApplication.installTranslator(self.translator)
101
102
103
104
105    # Declare instance attributes
106    self.actions = []
107    self.pluginIsActive = False
108
109
110
111    self.menu = QMenu("&CPLUS")
112    self.menu.setIcon(QIcon(ICON_PATH))
113
114
115
116    self.raster_menu = self.iface.rasterMenu()
117    self.raster_menu.addMenu(self.menu)
118
119
120
121    self.toolbar = self.iface.addToolBar("Open CPLUS")
122    self.toolbar.setObjectName("CPLUS")
123    self.toolButton = QToolButton()
124    self.toolButton.setMenu(QMenu())
125    self.toolButton.setPopupMode(QToolButton.MenuButtonPopup)
126    self.toolBtnAction = self.toolbar.addWidget(self.toolButton)
127    self.actions.append(self.toolBtnAction)
128
129
130
131    create_priority_layers()
```

```
initialize_model_settings()

self.main_widget = QgisCplusMain(
    iface=self.iface, parent=self.iface mainWindow()
)

self.options_factory = None
```

add_action

 **Code:**

```
add_action(
    icon_path,
    text,
    callback,
    enabled_flag=True,
    add_to_menu=True,
    add_to_web_menu=True,
    add_to_toolbar=True,
    set_as_default_action=False,
    status_tip=None,
    whats_this=None,
    parent=None,
)
```

Add a toolbar icon to the toolbar.

Parameters:

Name	Type	Description	Default
icon_path	str	Path to the icon for this action	required
text	str	Text that should be shown in menu items for this action	required
callback	function	Function to be called when the action is triggered	required
enabled_flag	bool	A flag indicating if the action should be enabled	True
add_to_menu	bool	Flag indicating whether the action should also be added to the menu	True
add_to_web_menu	bool	Flag indicating whether the action should also be added to the web menu	True
add_to_toolbar	bool	Flag indicating whether the action should also be added to the toolbar	True
set_as_default_action	bool	Flag indicating whether the action is the default action	False
status_tip	str	Optional text to show in a popup when mouse pointer hovers over the action	None
parent	QWidget	Parent widget for the new action	None
whats_this	str	Optional text to show in the status bar when the mouse pointer hovers over the action	None

Returns:

Type	Description
------	-------------

QAction	The action that was created
---------	-----------------------------

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/main.py`

134
135  **Code:**

136
137
138
139
140
141 def add_action(
142 self,
143 icon_path,
144 text,
145 callback,
146 enabled_flag=True,
147 add_to_menu=True,
148 add_to_web_menu=True,
149 add_to_toolbar=True,
150 set_as_default_action=False,
151 status_tip=None,
152 whats_this=None,
153 parent=None,
154):
155 """Add a toolbar icon to the toolbar.
156
157 :param icon_path: Path to the icon for this action
158 :type icon_path: str
159
160 :param text: Text that should be shown in menu items for this action
161 :type text: str
162
163 :param callback: Function to be called when the action is triggered
164 :type callback: function
165
166 :param enabled_flag: A flag indicating if the action should be enabled
167 :type enabled_flag: bool
168
169 :param add_to_menu: Flag indicating whether the action should also be added
170 :type add_to_menu: bool
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190

```
191 :param add_to_web_menu: Flag indicating whether the action should also be added to the web menu
192 :type add_to_web_menu: bool
193
194 :param add_to_toolbar: Flag indicating whether the action should also be added to the toolbar
195 :type add_to_toolbar: bool
196
197 :param set_as_default_action: Flag indicating whether the action is the default for its category
198 :type set_as_default_action: bool
199
200 :param status_tip: Optional text to show in a popup when mouse pointer hovers over the action
201 :type status_tip: str
202
203 :param parent: Parent widget for the new action
204 :type parent: QWidget
205
206 :param whats_this: Optional text to show in the status bar when the mouse pointer hovers over the action
207 :type whats_this: str
208
209
210
211
212
213
214
215
216

    :returns: The action that was created
    :rtype: QAction
    """
    icon = QIcon(icon_path)
    action = QAction(icon, text, parent)
    action.triggered.connect(callback)
    action.setEnabled(enabled_flag)

    if status_tip is not None:
        action.setStatusTip(status_tip)

    if whats_this is not None:
        action.setWhatsThis(whats_this)

    if add_to_menu:
        self.menu.addAction(action)
```

```
# If we want to read this
# if add_to_web_menu:
#     self iface.addPluginToWebMenu(self.menu, action)

if add_to_toolbar:
    self.toolButton.menu().addAction(action)

if set_as_default_action:
    self.toolButton.setDefaultAction(action)

if add_to_menu:
    self.menu.addAction(action)

self.actions.append(action)

return action
```

initGui

 **Code:**

initGui()

Create the menu entries and toolbar icons inside the QGIS GUI.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/main.py`

```
218
219  Code:
220
221
222
223
224
225     def initGui(self):
226         """Create the menu entries and toolbar icons inside the QGIS GUI."""
227         self.addAction(
228             QIcon(iconPath),
229             text=self.tr("CPLUS"),
230             callback=self.run,
231             parent=self.iface mainWindow(),
232             set_as_default_action=True,
233         )
234
235         self.addAction(
236             os.path.join(os.path.dirname(__file__), "icons", "settings.svg"),
237             text=self.tr("Settings"),
238             callback=self.run_settings,
239             parent=self.iface mainWindow(),
240             status_tip=self.tr("CPLUS Settings"),
241         )
242
243         self.addAction(
244             os.path.join(
245                 os.path.dirname(__file__), "icons", "mActionHelpContents_green.svg"
246             ),
247             text=self.tr("Help"),
248             callback=self.open_help,
249             parent=self.iface mainWindow(),
250             status_tip=self.tr("CPLUS Help"),
251         )
252
253         self.addAction(
254             os.path.join(os.path.dirname(__file__), "icons", "info_green.svg"),
255             text=self.tr("About"),
256         )
```

```
        callback=self.open_about,
        parent=self iface mainWindow(),
        status_tip=self.tr("CPLUS About"),
    )

    # Initialize default report settings
    initialize_report_settings()

    # Adds the settings to the QGIS options panel
    self.options_factory = CplusOptionsFactory()
    self iface.registerOptionsWidgetFactory(self.options_factory)

    # Register custom layout items
    self.register_layout_items()

    # Register custom report variables when a layout is opened
    self iface.layoutDesignerOpened.connect(self.on_layout_designer_opened)
```

onClosePlugin

 **Code:**

onClosePlugin()

Cleanup necessary items here when plugin widget is closed.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/main.py

267
268
269**Code:**

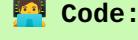
```
def onClosePlugin(self):  
    """Cleanup necessary items here when plugin widget is closed."""  
    self.pluginIsActive = False
```

on_layout_designer_opened**Code:**

```
on_layout_designer_opened(designer)
```

Register custom report variables in a print layout only.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/main.py

299
300
301
302
303
304**Code:**

```
def on_layout_designer_opened(self, designer: QgsLayoutDesignerInterface):  
    """Register custom report variables in a print layout only."""  
    layout_type = designer.masterLayout().layoutType()  
    if layout_type == QgsMasterLayoutInterface.PrintLayout:  
        layout = designer.layout()  
        report_manager.register_variables(layout)
```

open_about **Code:**`open_about()`

Opens the about documentation for the plugin in a browser

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/main.py`

322
323
324

 **Code:**

```
def open_about(self):
    """Opens the about documentation for the plugin in a browser"""
    open_documentation(ABOUT_DOCUMENTATION_SITE)
```

open_help **Code:**`open_help()`

Opens documentation home page for the plugin in a browser

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/main.py

318
319
320

 **Code:**

```
def open_help(self):
    """Opens documentation home page for the plugin in a browser"""
    open_documentation(DOCUMENTATION_SITE)
```

register_layout_items

 **Code:**

```
register_layout_items()
```

Register custom layout items.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/main.py

```
306
307  Code:
308
309
310
311
312
313     def register_layout_items(self):
314         """Register custom layout items."""
315         # Register map layout item
316         QgsApplication.layoutItemRegistry().addLayoutItemType(
317             CplusMapRepeatItemLayoutItemMetadata()
318         )
319
320         # Register map GUI metadata
321         item_gui_registry = QgsGui.layoutItemGuiRegistry()
322         map_item_gui_metadata = CplusMapLayoutItemGuiMetadata()
323         item_gui_registry.addLayoutItemGuiMetadata(map_item_gui_metadata)
```

run

 **Code:**

run()

Creates the main widget for the plugin.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/main.py

```
282
283  Code:
284
285
286
287
288
289     def run(self):
290         """Creates the main widget for the plugin."""
291         if self.main_widget is None:
292             self.main_widget = QgisCplusMain(
293                 iface=self iface, parent=self iface mainWindow()
294             )
295
296             self iface.addDockWidget(Qt.RightDockWidgetArea, self main_w.
297             self.main_widget.show()
298
299
300             if not self.pluginIsActive:
301                 self.pluginIsActive = True
```

run_settings

 **Code:**

```
run_settings()
```

Options the CPLUS settings in the QGIS options dialog.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/main.py

295
296
297

 **Code:**

```
def run_settings(self):
    """Options the CPLUS settings in the QGIS options dialog."""
    self.iface.showOptionsDialog(currentPage=OPTIONS_TITLE)
```

tr

 **Code:**

```
tr(message)
```

Get the translation for a string using Qt translation API. We implement this ourselves since we do not inherit QObject.

Parameters:

Name	Type	Description	Default
message	str	String for translation	required

Returns:

Type	Description
QString	Translated version of the message

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/main.py

```
121
122  Code:
123
124
125
126
127
128
129
130
131
132
```

```
def tr(self, message) -> str:
    """Get the translation for a string using Qt translation API.
    We implement this ourselves since we do not inherit QObject.

    :param message: String for translation
    :type message: str

    :returns: Translated version of the message
    :rtype: QString
    """
    # noinspection PyTypeChecker,PyArgumentList,PyCallByClass
    return QCoreApplication.translate("CPLUS", message)
```

unload

```
 Code:
```

```
unload()
```

Removes the plugin menu item and icon from QGIS GUI.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/main.py

```
271
272  Code:
273
274
275
276
277
278     def unload(self):
279         """Removes the plugin menu item and icon from QGIS GUI."""
280         try:
281             for action in self.actions():
282                 self.iface.removePluginMenu(self.tr("&CPLUS"), action)
283                 self.iface.removePluginWebMenu(self.tr("&CPLUS"), action)
284                 self.iface.removeToolBarIcon(action)
285
286         except Exception as e:
287             pass
```

Configuration

Handles storage and retrieval of the plugin QgsSettings.

ScenarioSettings dataclass

Bases: Scenario

Plugin Scenario settings.

from_qgs_settings classmethod

 **Code:**

```
from_qgs_settings(identifier, settings)
```

Reads QGIS settings and parses them into a scenario settings instance with the respective settings values as properties.

Parameters:

Name	Type	Description	Default
identifier	str	Scenario identifier	required
settings	QgsSettings	Scenario identifier	required

Returns:

Type	Description
ScenarioSettings	Scenario settings object

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
73
74      Code:
75
76
77
78
79
80 @classmethod
81 def from_qgs_settings(cls, identifier: str, settings: QgsSettings):
82     """Reads QGIS settings and parses them into a scenario
83     settings instance with the respective settings values as properties.
84
85     :param identifier: Scenario identifier
86     :type identifier: str
87
88     :param settings: Scenario identifier
89     :type settings: QgsSettings
90
91     :returns: Scenario settings object
92     :rtype: ScenarioSettings
93     """
94
95     return cls(
96         uuid=uuid.UUID(identifier),
97         name=settings.value("name", None),
98         description=settings.value("description", None),
99     )
```

get_scenario_extent classmethod

 **Code:**

```
get_scenario_extent()
```

Fetches Scenario extent from the passed scenario settings.

Returns:

Type	Description
SpatialExtent	Spatial extent instance extent

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
94
95
96     🤖 Code:
97
98
99
100
101 @classmethod
102
103 def get_scenario_extent(cls):
104     """Fetches Scenario extent from
105     the passed scenario settings.
106
107
108
109
110
111     :returns: Spatial extent instance extent
112     :rtype: SpatialExtent
113     """
114
115     spatial_key = "extent/spatial"
116
117
118     with qgis_settings(spatial_key, cls) as settings:
119         bbox = settings.value("bbox", None)
120         spatial_extent = SpatialExtent(bbox=bbox)
121
122
123     return spatial_extent
```

Settings

Bases: **Enum**

Plugin settings names

SettingsManager

Bases: **QObject**

Manages saving/loading settings for the plugin in QgsSettings.

delete_all_scenarios

Code:

```
delete_all_scenarios()
```

Deletes all the plugin scenarios settings.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

355
356
357
358
359
360
361

```
def delete_all_scenarios(self):
    """Deletes all the plugin scenarios settings."""
    with qgis_settings(
        f'{self.BASE_GROUP_NAME}"/ {self.SCENARIO_GROUP_NAME}'
    ) as settings:
        for scenario_name in settings.childGroups():
            settings.remove(scenario_name)
```

delete_priority_group

Code:

```
delete_priority_group(identifier)
```

Removes priority group that match the passed identifier

Parameters:

Name	Type	Description	Default
identifier	str	Priority group identifier required	

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
676
677
678  Code:
679
680
681
682
683     def delete_priority_group(self, identifier):
684         """Removes priority group that match the passed identifier
685
686         :param identifier: Priority group identifier
687         :type identifier: str
688         """
689
690         with qgis_settings(
691             f"{self.BASE_GROUP_NAME}/" f"{self.PRIORITY_GROUP_NAME}/"
692         ) as settings:
693             for priority_group in settings.childGroups():
694                 if str(priority_group) == str(identifier):
695                     settings.remove(priority_group)
```

delete_priority_groups

 **Code:**

```
delete_priority_groups()
```

Deletes all the plugin priority groups settings.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
689
690
691  Code:
692
693
694
695

def delete_priority_groups(self):
    """Deletes all the plugin priority groups settings."""
    with qgis_settings(
        f"{self.BASE_GROUP_NAME}"/ f"{self.PRIORITY_GROUP_NAME}"
    ) as settings:
        for priority_group in settings.childGroups():
            settings.remove(priority_group)
```

delete_priority_layer

```
 Code:

delete_priority_layer(identifier)
```

Removes priority layer that match the passed identifier

Parameters:

Name	Type	Description	Default
identifier	str	Priority layer identifier	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
565
566
567  Code:
568
569
570
571
572     def delete_priority_layer(self, identifier):
573         """Removes priority layer that match the passed identifier
574
575         :param identifier: Priority layer identifier
576         :type identifier: str
577
578         """
579
580         with qgis_settings(
581             f"{self.BASE_GROUP_NAME}"/ f"{self.PRIORITY_LAYERS_GROUP}"
582         ) as settings:
583             for priority_layer in settings.childGroups():
584                 if str(priority_layer) == str(identifier):
585                     settings.remove(priority_layer)
```

delete_priority_layers

 **Code:**

```
delete_priority_layers()
```

Deletes all the plugin priority weighting layers settings.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
557
558      Code:
559
560
561
562
563

def delete_priority_layers(self):
    """Deletes all the plugin priority weighting layers settings."""
    with qgis_settings(
        f"{self.BASE_GROUP_NAME}"/ f"{self.PRIORITY_LAYERS_GROUP_NAME}"
    ) as settings:
        for priority_layer in settings.childGroups():
            settings.remove(priority_layer)
```

delete_settings

```
Code:
delete_settings()
```

Deletes the all the plugin settings.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
238
239      Code:
240

def delete_settings(self):
    """Deletes the all the plugin settings."""
    self.settings.remove(f"{self.BASE_GROUP_NAME}")
```

find_group_by_name

Code:

```
find_group_by_name(name)
```

Finds a priority group setting inside the plugin QgsSettings by name.

Parameters:

Name	Type	Description	Default
name	str	Name of the group	required

Returns:

Type	Description
typing.Dict	Priority group

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

594
595
596 **Code:**
597
598
599
600
601 def find_group_by_name(self, name) -> typing.Dict:
602 """Finds a priority group setting inside the plugin QgsSettings by name.
603
604 :param name: Name of the group
605 :type name: str
606
607 :returns: Priority group
608 :rtype: typing.Dict
609 """
610
611
612
613
614
615
616
617 found_id = None

 with qgis_settings(
 f"{self.BASE_GROUP_NAME}"/ f"{self.PRIORITY_GROUP_NAME}"
) as settings:
 for group_id in settings.childGroups():
 group_settings_key = self._get_priority_groups_settings_base(group_id)
 with qgis_settings(group_settings_key) as group_settings_key:
 group_name = group_settings_key.value("name")
 if group_name == name:
 found_id = uuid.UUID(group_id)
 break

 return self.get_priority_group(found_id)

find_implementation_model_by_name

Code:

```
find_implementation_model_by_name(name)
```

Finds an implementation model setting inside the plugin QgsSettings that equals or matches the name.

Parameters:

Name	Type	Description	Default
name	str	Implementation model name required	

Returns:

Type	Description
ImplementationModel	Implementation model

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
944
945  Code:
946
947
948
949
950
951     def find_implementation_model_by_name(self, name) -> typing.Dict
952         """Finds an implementation model setting inside
953         the plugin QgsSettings that equals or matches the name.
954
955         :param name: Implementation model name
956         :type name: str
957
958         :returns: Implementation model
959         :rtype: ImplementationModel
960         """
961
962         for model in self.get_all_implementation_models():
963             model_name = model.name
964             trimmed_name = model_name.replace(" ", "_")
965             if model_name == name or model_name in name or trimmed_name == name:
966                 return model
967
968
969     return None
```

find_layer_by_name



```
find_layer_by_name(name)
```

Finds a priority layer setting inside the plugin QgsSettings by name.

Parameters:

Name	Type	Description	Default
name	str	Priority layers identifier required	

Returns:**Type Description****dict** Priority layers dict

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

457
458
459  **Code:**
460
461
462
463
464
465 def find_layer_by_name(self, name) -> typing.Dict:
466 """Finds a priority layer setting inside
467 the plugin QgsSettings by name.
468
469 :param name: Priority layers identifier
470 :type name: str
471
472 :returns: Priority layers dict
473 :rtype: dict
474 """
475
476 found_id = None
477 with qgis_settings(
478 f"{self.BASE_GROUP_NAME}"/ f"{self.PRIORITY_LAYERS_GROUP}_") as settings:
479 for layer_id in settings.childGroups():
 layer_settings_key = self._get_priority_layers_setting_key(layer_id)
 with qgis_settings(layer_settings_key) as layer_settings:
 layer_name = layer_settings.value("name")
 if layer_name == name:
 found_id = uuid.UUID(layer_id)
 break

 return self.get_priority_layer(found_id) if found_id is not None else None

find_layers_by_group

Code:

```
find_layers_by_group(group)
```

Finds priority layers inside the plugin QgsSettings that contain the passed group.

Parameters:

Name	Type	Description	Default
group	str	Priority group name	required

Returns:

Type	Description
list	Priority layers list

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

481
482
483  **Code:**
484
485
486
487
488 def find_layers_by_group(self, group) -> typing.List:
489 """Finds priority layers inside the plugin QgsSettings
490 that contain the passed group.
491
492 :param group: Priority group name
493 :type group: str
494
495 :returns: Priority layers list
496 :rtype: list
497 """
498
499 layers = []
500 with qgis_settings(
501 f"{self.BASE_GROUP_NAME}"/ f"{self.PRIORITY_LAYERS_GROUP_NAME}"
502) as settings:
503 for layer_id in settings.childGroups():
504 priority_layer_settings = self._get_priority_layers_settings_base(
505 layer_id
506)
507 with qgis_settings(priority_layer_settings) as priority_settings:
508 groups_key = f"{priority_layer_settings}/groups"
509
510 with qgis_settings(groups_key) as groups_settings:
511 for name in groups_settings.childGroups():
512 group_settings_key = f"{groups_key}/{name}"
513 with qgis_settings(group_settings_key) as group_setting:
514 if group == group_setting.value("name"):
515 layers.append(self.get_priority_layer(layer_id))
516
517 return layers

find_settings**Code:**

```
find_settings(name)
```

Returns the plugin setting keys from the plugin root group that matches the passed name

Parameters:

Name	Type	Description	Default
name	str	Setting name to search for	required

Returns:

Type	Description
list	List of the matching settings names

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
212
213
214  Code:
215
216
217
218
219     def find_settings(self, name):
220         """Returns the plugin setting keys from the
221             plugin root group that matches the passed name
222
223         :param name: Setting name to search for
224         :type name: str
225
226         :returns result: List of the matching settings names
227         :rtype result: list
228         """
229
230
231         result = []
232         with qgis_settings(f"{self.BASE_GROUP_NAME}") as settings:
233             for settings_name in settings.childKeys():
234                 if name in settings_name:
235                     result.append(settings_name)
236
237     return result
```

get_all_implementation_models

 **Code:**

```
get_all_implementation_models()
```

Get all the implementation model objects stored in settings.

Returns:

Type Description

list Returns all the implementation model objects.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
962
963
964
965
966
967
968
969     def get_all_implementation_models(self) -> typing.List[Implementation
970         """Get all the implementation model objects stored in settings.
971
972         :returns: Returns all the implementation model objects.
973         :rtype: list
974         """
975
976         implementation_models = []
977
978
979
    implementation_model_root = self._get_implementation_model_setting()
    with qgis_settings(implementation_model_root) as settings:
        keys = settings.childKeys()
        for k in keys:
            implementation_model = self.get_implementation_model(k)
            if implementation_model is not None:
                implementation_models.append(implementation_model)

    return sorted(implementation_models, key=lambda imp_model: imp_model.name)
```

get_all_ncs_pathways

Code:

```
get_all_ncs_pathways()
```

Get all the NCS pathway objects stored in settings.

Returns:**Type Description**

`list` Returns all the NCS pathway objects.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py`

```
768  
769  
770     🤖 Code:  
771  
772  
773  
774  
775     def get_all_ncs_pathways(self) -> typing.List[NcsPathway]:  
776         """Get all the NCS pathway objects stored in settings.  
777  
778             :returns: Returns all the NCS pathway objects.  
779             :rtype: list  
780             """  
781  
782         ncs_pathways = []  
783  
784  
785
```

```
        ncs_root = self._get_ncs_pathway_settings_base()  
  
        with qgis_settings(ncs_root) as settings:  
            keys = settings.childKeys()  
            for k in keys:  
                ncs_pathway = self.get_ncs_pathway(k)  
                if ncs_pathway is not None:  
                    ncs_pathways.append(ncs_pathway)  
  
    return sorted(ncs_pathways, key=lambda ncs: ncs.name)
```

get_implementation_model

芜 **Code:**

```
get_implementation_model(implementation_model_uuid)
```

Gets an implementation model object matching the given unique identified.

Parameters:

Name	Type	Description	Default
implementation_model_uuid	str	Unique identifier for the implementation model object.	required

Returns:

Type	Description
ImplementationModel	Returns the implementation model object matching the given identifier else None if not found.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

902
903
904  **Code:**
905
906
907
908
909 def get_implementation_model(
910 self, implementation_model_uuid: str
911) -> typing.Union[ImplementationModel, None]:
912 """Gets an implementation model object matching the given unique
913 identified.
914
915 :param implementation_model_uuid: Unique identifier for the
916 implementation model object.
917 :type implementation_model_uuid: str
918
919 :returns: Returns the implementation model object matching the given
920 identifier else None if not found.
921 :rtype: ImplementationModel
922 """
923
924
925 implementation_model = None
926
927
928 implementation_model_root = self._get_implementation_model_settings_ba
929
930
931 with qgis_settings(implementation_model_root) as settings:
932 implementation_model = settings.value(implementation_model_uuid, N
933 ncs_uuids = []
934 if implementation_model is not None:
935 implementation_model_dict = {}
936 try:
937 implementation_model_dict = json.loads(implementation_mode
938 except json.JSONDecodeError:
939 log("Implementation model JSON is invalid.")
940
941
942 if PATHWAYS_ATTRIBUTE in implementation_model_dict:
943 ncs_uuids = implementation_model_dict[PATHWAYS_ATTRIBUTE]

```
implementation_model = create_implementation_model(  
    implementation_model_dict  
)  
if implementation_model is not None:  
    for ncs_uuid in ncs_uuids:  
        ncs = self.get_ncs_pathway(ncs_uuid)  
        if ncs is not None:  
            implementation_model.add_ncs_pathway(ncs)  
  
return implementation_model
```

get_ncs_pathway

Code:

```
get_ncs_pathway(ncs_uuid)
```

Gets an NCS pathway object matching the given unique identified.

Parameters:

Name	Type	Description	Default
ncs_uuid	str	Unique identifier for the NCS pathway object.required	

Returns:

Type	Description
NcsPathway	Returns the NCS pathway object matching the given identifier else None if not found.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
724
725  Code:
726
727
728
729
730
731     def get_ncs_pathway(self, ncs_uuid: str) -> typing.Union[NcsPathway, None]:
732         """Gets an NCS pathway object matching the given unique identified.
733
734         :param ncs_uuid: Unique identifier for the NCS pathway object.
735         :type ncs_uuid: str
736
737         :returns: Returns the NCS pathway object matching the given
738         identifier else None if not found.
739         :rtype: NcsPathway
740
741         """
742
743         ncs_pathway = None
744
745         ncs_dict = self.get_ncs_pathway_dict(ncs_uuid)
746         if len(ncs_dict) == 0:
747             return None
748
749         ncs_pathway = create_ncs_pathway(ncs_dict)
750
751         return ncs_pathway
```

get_ncs_pathway_dict



```
get_ncs_pathway_dict(ncs_uuid)
```

Gets an NCS pathway attribute values as a dictionary.

Parameters:

Name	Type	Description	Default
ncs_uuid	str	Unique identifier for the NCS pathway object.required	

Returns:

Type	Description
dict	Returns the NCS pathway attribute values matching the given identifier else an empty dictionary if not found.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

744
745 Code:
746
747
748
749
750
751 def get_ncs_pathway_dict(self, ncs_uuid: str) -> dict:
752 """Gets an NCS pathway attribute values as a dictionary.
753
754
755 :param ncs_uuid: Unique identifier for the NCS pathway object.
756 :type ncs_uuid: str
757
758 :returns: Returns the NCS pathway attribute values matching the given
759 identifier else an empty dictionary if not found.
760 :rtype: dict
761 """
762
763
764 ncs_pathway_dict = {}
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
17010
17011
17012
17013
17014
17015
17016
17017
17018
17019
17020
17021
17022
17023
17024
17025
17026
17027
17028
17029
17030
17031
17032
17033
17034
17035
17036
17037
17038
17039
17040
17041
17042
17043
17044
17045
17046
17047
17048
17049
17050
17051
17052
17053
17054
17055
17056
17057
17058
17059
17060
17061
17062
17063
17064
17065
17066
17067
17068
17069
17070
17071
17072
17073
17074
17075
17076
17077
17078
17079
17080
17081
17082
17083
17084
17085
17086
17087
17088
17089
17090
17091
17092
17093
17094
17095
17096
17097
17098
17099
170100
170101
170102
170103
170104
170105
170106
170107
170108
170109
170110
170111
170112
170113
170114
170115
170116
170117
170118
170119
170120
170121
170122
170123
170124
170125
170126
170127
170128
170129
170130
170131
170132
170133
170134
170135
170136
170137
170138
170139
170140
170141
170142
170143
170144
170145
170146
170147
170148
170149
170150
170151
170152
170153
170154
170155
170156
170157
170158
170159
170160
170161
170162
170163
170164
170165
170166
170167
170168
170169
170170
170171
170172
170173
170174
170175
170176
170177
170178
170179
170180
170181
170182
170183
170184
170185
170186
170187
170188
170189
170190
170191
170192
170193
170194
170195
170196
170197
170198
170199
170200
170201
170202
170203
170204
170205
170206
170207
170208
170209
170210
170211
170212
170213
170214
170215
170216
170217
170218
170219
170220
170221
170222
170223
170224
170225
170226
170227
170228
170229
170230
170231
170232
170233
170234
170235
170236
170237
170238
170239
170240
170241
170242
170243
170244
170245
170246
170247
170248
170249
170250
170251
170252
170253
170254
170255
170256
170257
170258
170259
170260
170261
170262
170263
170264
170265
170266
170267
170268
170269
170270
170271
170272
170273
170274
170275
170276
170277
170278
170279
170280
170281
170282
170283
170284
170285
170286
170287
170288
170289
170290
170291
170292
170293
170294
170295
170296
170297
170298
170299
170300
170301
170302
170303
170304
170305
170306
170307
170308
170309
170310
170311
170312
170313
170314
170315
170316
170317
170318
170319
170320
170321
170322
170323
170324
170325
170326
170327
170328
170329
170330
170331
170332
170333
170334
170335
170336
170337
170338
170339
170340
170341
170342
170343
170344
170345
170346
170347
170348
170349
170350
170351
170352
170353
170354
170355
170356
170357
170358
170359
170360
170361
170362
170363
170364
170365
170366
170367
170368
170369
170370
170371
170372
170373
170374
170375
170376
170377
170378
170379
170380
170381
170382
170383
170384
170385
170386
170387
170388
170389
170390
170391
170392
170393
170394
170395
170396
170397
170398
170399
170400
170401
170402
170403
170404
170405
170406
170407
170408
170409
170410
170411
170412
170413
170414
170415
170416
170417
170418
170419
170420
170421
170422
170423
170424
170425
170426
170427
170428
170429
170430
170431
170432
170433
170434
170435
170436
170437
170438
170439
170440
170441
170442
170443
170444
170445
170446
170447
170448
170449
170450
170451
170452
170453
170454
170455
170456
170457
170458
170459
170460
170461
170462
170463
170464
170465
170466
170467
170468
170469
170470
170471
170472
170473
170474
170475
170476
170477
170478
170479
170480
170481
170482
170483
170484
170485
170486
170487
170488
170489
170490
170491
170492
170493
170494
170495
170496
170497
170498
170499
170500
170501
170502
170503
170504
170505
170506
170507
170508
170509
170510
170511
170512
170513
170514
170515
170516
170517
170518
170519
170520
170521
170522
170523
170524
170525
170526
170527
170528
170529
170530
170531
170532
170533
170534
170535
170536
170537
170538
170539
170540
170541
170542
170543
170544
170545
170546
170547
170548
170549
170550
170551
170552
170553
170554
170555
170556
170557
170558
170559
170560
170561
170562
170563
170564
170565
170566
170567
170568
170569
170570
170571
170572
170573
170574
170575
170576
170577
170578
170579
170580
170581
170582
170583
170584
170585
170586
170587
170588
170589
170590
170591
170592
170593
170594
170595
170596
170597
170598
170599
170600
170601
170602
170603
170604
170605
170606
170607
170608
170609
170610
170611
170612
170613
170614
170615
170616
170617
170618
170619
170620
170621
170

get_priority_group

Code:

```
get_priority_group(identifier)
```

Retrieves the priority group that matches the passed identifier.

Parameters:

Name	Type	Description	Default
identifier	str	Priority group identifier	required

Returns:

Type	Description
typing.Dict	Priority group

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
619
620
621  Code:
622
623
624
625
626     def get_priority_group(self, identifier) -> typing.Dict:
627         """Retrieves the priority group that matches the passed identifier.
628
629         :param identifier: Priority group identifier
630         :type identifier: str
631
632         :returns: Priority group
633         :rtype: typing.Dict
634
635         """
636
637
638         if identifier is None:
639             return None
640
641
642         settings_key = self._get_priority_groups_settings_base(identifier)
643         with qgis_settings(settings_key) as settings:
644             priority_group = {"uuid": identifier}
645             priority_group["name"] = settings.value("name")
646             priority_group["value"] = settings.value("value")
647             priority_group["description"] = settings.value("description")
648
649         return priority_group
```

get_priority_groups



```
get_priority_groups()
```

Gets all the available priority groups in the plugin.

Returns:**Type Description****list** List of the priority groups instances

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

640
641
642 **Code:**
643
644
645
646
647 def get_priority_groups(self) -> typing.List[typing.Dict]:
648 """Gets all the available priority groups in the plugin.
649
650
651 :returns: List of the priority groups instances
652 :rtype: list
653 """
654
655 priority_groups = []
656 with qgis_settings(
657 f"{self.BASE_GROUP_NAME}"/ f"{self.PRIORITY_GROUP_NAME}"
658) as settings:
659 for uuid in settings.childGroups():
660 priority_layer_settings = self._get_priority_groups_
661 with qgis_settings(priority_layer_settings) as priori
662 group = {
663 "uuid": uuid,
664 "name": priority_settings.value("name"),
665 "value": priority_settings.value("value"),
666 "description": priority_settings.value("desc
667 }
668 priority_groups.append(group)
669 return priority_groups

get_priority_layer

 **Code:**

```
get_priority_layer(identifier)
```

Retrieves the priority layer that matches the passed identifier.

Parameters:

Name	Type	Description	Default
identifier	uuid.UUID	Priority layers identifier required	

Returns:

Type	Description
dict	Priority layer dict

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

 **Code:**

```
378
379
380
381
382
383
384
385 def get_priority_layer(self, identifier) -> typing.Dict:
386     """Retrieves the priority layer that matches the passed identifier.
387
388     :param identifier: Priority layers identifier
389     :type identifier: uuid.UUID
390
391     :returns: Priority layer dict
392     :rtype: dict
393
394     """
395     priority_layer = None
396
397     settings_key = self._get_priority_layers_settings_base(identifier)
398     with qgis_settings(settings_key) as settings:
399         groups_key = f"{settings_key}/groups"
400         groups = []
401
402         if len(settings.childKeys()) <= 0:
403             return priority_layer
404
405         with qgis_settings(groups_key) as groups_settings:
406             for name in groups_settings.childGroups():
407                 group_settings_key = f"{groups_key}/{name}"
408                 with qgis_settings(group_settings_key) as group_settings:
409                     stored_group = {}
410                     stored_group["uuid"] = group_settings.value("uuid")
411                     stored_group["name"] = group_settings.value("name")
412                     stored_group["value"] = group_settings.value("value")
413                     groups.append(stored_group)
414
415
416         priority_layer = {"uuid": str(identifier)}
417         priority_layer["name"] = settings.value("name")
```

```
priority_layer["description"] = settings.value("description")
priority_layer["path"] = settings.value("path")
priority_layer["selected"] = settings.value("selected", type=bool)
priority_layer["user_defined"] = settings.value(
    "user_defined", defaultValue=True, type=bool
)
priority_layer["groups"] = groups
return priority_layer
```

get_priority_layers

Code:

```
get_priority_layers()
```

Gets all the available priority layers in the plugin.

Returns:

Type Description

list Priority layers list

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py`

418
419
420  **Code:**
421
422
423
424
425 def get_priority_layers(self) -> typing.List:
426 """Gets all the available priority layers in the plugin.
427
428 :returns: Priority layers list
429 :rtype: list
430 """
431 priority_layer_list = []
432 with qgis_settings(
433 f"{self.BASE_GROUP_NAME}"/ f"{self.PRIORITY_LAYERS_GROUP_}
434) as settings:
435 for uuid in settings.childGroups():
436 priority_layer_settings = self._get_priority_layers_
437 with qgis_settings(priority_layer_settings) as priori
438 groups_key = f'{priority_layer_settings}/groups'
439 groups = []
440
441 with qgis_settings(groups_key) as groups_setting:
442 for name in groups_settings.childGroups():
443 group_settings_key = f'{groups_key}/{name}'
444 with qgis_settings(group_settings_key) as
445 stored_group = {}
446 stored_group["uuid"] = group_setting.
447 stored_group["name"] = group_setting.
448 stored_group["value"] = group_setting.
449 groups.append(stored_group)
450
451 layer = {
452 "uuid": uuid,
453 "name": priority_settings.value("name"),
454 "description": priority_settings.value("desci"),
455 "path": priority_settings.value("path"),
456 "selected": priority_settings.value("selected")

```
        "user_defined": priority_settings.value(  
            "user_defined", defaultValue=True, type=  
        ),  
        "groups": groups,  
    }  
    priority_layer_list.append(layer)  
return priority_layer_list
```

get_scenario**Code:**

```
get_scenario(scenario_id)
```

Retrieves the first scenario that matched the passed scenario id.

Parameters:

Name	Type	Description	Default
scenario_id	str	Scenario id	required

Returns:

Type	Description
ScenarioSettings	Scenario settings instance

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

309
310 **Code:**
311
312
313
314
315
316 def get_scenario(self, scenario_id):
317 """Retrieves the first scenario that matched the passed scenario id.
318
319 :param scenario_id: Scenario id
320 :type scenario_id: str
321
322 :returns: Scenario settings instance
323 :rtype: ScenarioSettings
324
325 """
326
327
328
329
330
331
332 result = []
333 with qgis_settings(
334 f"{self.BASE_GROUP_NAME}"/ f"{self.SCENARIO_GROUP_NAME}"
335) as settings:
336 for uuid in settings.childGroups():
337 scenario_settings_key = self._get_scenario_settings_base(uuid)
338 with qgis_settings(scenario_settings_key) as scenario_settings:
339 scenario = ScenarioSettings.from_qgs_settings(
340 uuid, scenario_settings
341)
342 if scenario.id == scenario_id:
343 return scenario
344
345 return None

get_scenarios **Code:**

```
get_scenarios()
```

Gets all the available scenarios settings in the plugin.

Returns:**Type Description**

list List of the scenario settings instances

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
333
334  Code:
335
336
337
338
339
340     def get_scenarios(self):
341         """Gets all the available scenarios settings in the plugin.
342
343         :returns: List of the scenario settings instances
344         :rtype: list
345
346         """
347
348         result = []
349         with qgis_settings(
350             f"{self.BASE_GROUP_NAME}"/" {self.SCENARIO_GROUP_NAME}"
351         ) as settings:
352             for uuid in settings.childGroups():
353                 scenario_settings_key = self._get_scenario_settings_b
354                 with qgis_settings(scenario_settings_key) as scenario_
355                     scenario = ScenarioSettings.from_qgs_settings(
356                         uuid, scenario_settings
357                     )
358                     scenario.extent = self.get_scenario_
359                     result.append(
360                         ScenarioSettings.from_qgs_settings(uuid, scenar
361                     )
362
363         return result
```

get_value

 **Code:**

```
get_value(name, default=None, setting_type=None)
```

Gets value of the setting with the passed name.

Parameters:

Name	Type	Description	Default
name	str	Name of setting key	required
default	Any	Default value returned when the setting key does not exist	None
setting_type	Any	Type of the store setting	None

Returns:

Type Description

Any Value of the setting

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py](#)

191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210

 **Code:**

```
def get_value(self, name: str, default=None, setting_type=None):
    """Gets value of the setting with the passed name.

    :param name: Name of setting key
    :type name: str

    :param default: Default value returned when the setting key does not exist
    :type default: Any

    :param setting_type: Type of the store setting
    :type setting_type: Any

    :returns: Value of the setting
    :rtype: Any
    """

    if setting_type:
        return self.settings.value(
            f"{self.BASE_GROUP_NAME}/{name}", default, setting_type
        )
    return self.settings.value(f"{self.BASE_GROUP_NAME}/{name}", default)
```

remove **Code:**

```
remove(name)
```

Remove the setting with the specified name.

Parameters:

Name	Type	Description	Default
------	------	-------------	---------

name	str	Name of the setting key required	
------	-----	----------------------------------	--

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
230  
231  
232  
233  
234  
235  
236
```

```
 Code:  
  
def remove(self, name):  
    """Remove the setting with the specified name.  
  
    :param name: Name of the setting key  
    :type name: str  
    """  
  
    self.settings.remove(f"{self.BASE_GROUP_NAME}/{name}")
```

remove_implementation_model **Code:**

```
remove_implementation_model(implementation_model_uuid)
```

Removes an implementation model settings entry using the UUID.

Parameters:

Name	Type	Description	Default
implementation_model_uuid	str	Unique identifier of the implementation model entry to removed.	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
1021
1022
1023     🤖 Code:
1024
1025
1026
1027
1028
1029
def remove_implementation_model(self, implementation_model_uuid: str):
    """Removes an implementation model settings entry using the UUID.

    :param implementation_model_uuid: Unique identifier of the implementation model entry to removed.
    :type implementation_model_uuid: str
    """
    if self.get_implementation_model(implementation_model_uuid) is not None:
        self.remove(f"{self.IMPLEMENTATION_MODEL_BASE}/{implementation_model_uu
```

remove_ncs_pathway**Code:**

```
remove_ncs_pathway(ncs_uuid)
```

Removes an NCS pathway settings entry using the UUID.

Parameters:

Name	Type	Description	Default
ncs_uuid	str	Unique identifier of the NCS pathway entry to removed.	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
842
843     Code:
844
845
846
847
848
849
850
def remove_ncs_pathway(self, ncs_uuid: str):
    """Removes an NCS pathway settings entry using the UUID.

    :param ncs_uuid: Unique identifier of the NCS pathway entry
    to removed.
    :type ncs_uuid: str
    """
    if self.get_ncs_pathway(ncs_uuid) is not None:
        self.remove(f"{self.NCS_PATHWAY_BASE}/{ncs_uuid}")
```

save_implementation_model

Code:

```
save_implementation_model(implementation_model)
```

Saves an implementation model object serialized to a json string indexed by the UUID.

Parameters:

Name	Type	Description	Default
implementation_model	Union[ImplementationModel, dict]	Implementation required model object or attribute values in a dictionary which are then serialized to a JSON string.	

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

860
861  **Code:**
862
863
864
865
866
867 **def save_implementation_model(**
868 **self, implementation_model: typing.Union[ImplementationModel, dict]**
869 **):**
870 <="" data-bbox="238 304 948 336"/>
871 <="" data-bbox="238 356 948 404"/>
872 <="" data-bbox="238 404 777 418"/>
873 <="" data-bbox="238 418 488 432"/>
874 <="" data-bbox="238 432 448 446"/>
875 <="" data-bbox="238 446 278 459"/>
876 <="" data-bbox="238 459 848 473"/>
877 <="" data-bbox="238 473 908 487"/>
878 <="" data-bbox="238 487 908 501"/>
879 <="" data-bbox="238 501 777 515"/>
880 <="" data-bbox="238 515 488 529"/>
881 <="" data-bbox="238 529 388 543"/>
882 <="" data-bbox="238 543 278 556"/>
883 <="" data-bbox="238 556 828 570"/>
884 <="" data-bbox="238 570 838 584"/>
885 <="" data-bbox="238 584 777 598"/>
886 <="" data-bbox="238 598 777 612"/>
887 <="" data-bbox="238 612 878 626"/>
888 <="" data-bbox="238 626 908 640"/>
889 <="" data-bbox="238 640 908 654"/>
890 <="" data-bbox="238 654 908 668"/>
891 <="" data-bbox="238 668 908 682"/>
892 <="" data-bbox="238 682 908 696"/>
893 <="" data-bbox="238 696 908 710"/>
894 <="" data-bbox="238 710 908 724"/>
895 <="" data-bbox="238 724 908 738"/>
896 <="" data-bbox="238 738 908 752"/>
897 <="" data-bbox="238 752 908 766"/>
898 <="" data-bbox="238 766 908 780"/>
899 <="" data-bbox="238 780 908 794"/>
900 <="" data-bbox="238 794 908 808"/>

901 <="" data-bbox="238 529 908 543"/>
902 <="" data-bbox="238 543 908 556"/>
903 <="" data-bbox="238 556 908 569"/>
904 <="" data-bbox="238 569 908 583"/>
905 <="" data-bbox="238 583 908 597"/>
906 <="" data-bbox="238 597 908 611"/>
907 <="" data-bbox="238 611 908 625"/>
908 <="" data-bbox="238 625 908 639"/>
909 <="" data-bbox="238 639 908 653"/>
910 <="" data-bbox="238 653 908 667"/>
911 <="" data-bbox="238 667 908 681"/>
912 <="" data-bbox="238 681 908 695"/>
913 <="" data-bbox="238 695 908 709"/>
914 <="" data-bbox="238 709 908 723"/>
915 <="" data-bbox="238 723 908 737"/>
916 <="" data-bbox="238 737 908 751"/>
917 <="" data-bbox="238 751 908 765"/>
918 <="" data-bbox="238 765 908 779"/>
919 <="" data-bbox="238 779 908 793"/>
920 <="" data-bbox="238 793 908 807"/>
921 <="" data-bbox="238 807 908 821"/>
922 <="" data-bbox="238 821 908 835"/>
923 <="" data-bbox="238 835 908 849"/>
924 <="" data-bbox="238 849 908 863"/>
925 <="" data-bbox="238 863 908 877"/>
926 <="" data-bbox="238 877 908 891"/>
927 <="" data-bbox="238 891 908 905"/>
928 <="" data-bbox="238 905 908 919"/>
929 <="" data-bbox="238 919 908 933"/>
930 <="" data-bbox="238 933 908 947"/>
931 <="" data-bbox="238 947 908 961"/>
932 <="" data-bbox="238 961 908 975"/>
933 <="" data-bbox="238 975 908 989"/>
934 <="" data-bbox="238 989 908 1003"/>
935 <="" data-bbox="238 1003 908 1017"/>
936 <="" data-bbox="238 1017 908 1031"/>
937 <="" data-bbox="238 1031 908 1045"/>
938 <="" data-bbox="238 1045 908 1059"/>
939 <="" data-bbox="238 1059 908 1073"/>
940 <="" data-bbox="238 1073 908 1087"/>
941 <="" data-bbox="238 1087 908 1091"/>
942 <="" data-bbox="238 1091 908 1105"/>
943 <="" data-bbox="238 1105 908 1119"/>
944 <="" data-bbox="238 1119 908 1133"/>
945 <="" data-bbox="238 1133 908 1147"/>
946 <="" data-bbox="238 1147 908 1161"/>
947 <="" data-bbox="238 1161 908 1175"/>
948 <="" data-bbox="238 1175 908 1189"/>
949 <="" data-bbox="238 1189 908 1203"/>
950 <="" data-bbox="238 1203 908 1217"/>
951 <="" data-bbox="238 1217 908 1231"/>
952 <="" data-bbox="238 1231 908 1245"/>
953 <="" data-bbox="238 1245 908 1259"/>
954 <="" data-bbox="238 1259 908 1273"/>
955 <="" data-bbox="238 1273 908 1287"/>
956 <="" data-bbox="238 1287 908 1291"/>
957 <="" data-bbox="238 1291 908 1305"/>
958 <="" data-bbox="238 1305 908 1319"/>
959 <="" data-bbox="238 1319 908 1333"/>
960 <="" data-bbox="238 1333 908 1347"/>
961 <="" data-bbox="238 1347 908 1361"/>
962 <="" data-bbox="238 1361 908 1375"/>
963 <="" data-bbox="238 1375 908 1389"/>
964 <="" data-bbox="238 1389 908 1403"/>
965 <="" data-bbox="238 1403 908 1417"/>
966 <="" data-bbox="238 1417 908 1431"/>
967 <="" data-bbox="238 1431 908 1445"/>
968 <="" data-bbox="238 1445 908 1459"/>
969 <="" data-bbox="238 1459 908 1473"/>
970 <="" data-bbox="238 1473 908 1487"/>
971 <="" data-bbox="238 1487 908 1501"/>
972 <="" data-bbox="238 1501 908 1515"/>
973 <="" data-bbox="238 1515 908 1529"/>
974 <="" data-bbox="238 1529 908 1543"/>
975 <="" data-bbox="238 1543 908 1557"/>
976 <="" data-bbox="238 1557 908 1571"/>
977 <="" data-bbox="238 1571 908 1585"/>
978 <="" data-bbox="238 1585 908 1599"/>
979 <="" data-bbox="238 1599 908 1613"/>
980 <="" data-bbox="238 1613 908 1627"/>
981 <="" data-bbox="238 1627 908 1641"/>
982 <="" data-bbox="238 1641 908 1655"/>
983 <="" data-bbox="238 1655 908 1669"/>
984 <="" data-bbox="238 1669 908 1683"/>
985 <="" data-bbox="238 1683 908 1697"/>
986 <="" data-bbox="238 1697 908 1711"/>
987 <="" data-bbox="238 1711 908 1725"/>
988 <="" data-bbox="238 1725 908 1739"/>
989 <="" data-bbox="238 1739 908 1753"/>
990 <="" data-bbox="238 1753 908 1767"/>
991 <="" data-bbox="238 1767 908 1781"/>
992 <="" data-bbox="238 1781 908 1795"/>
993 <="" data-bbox="238 1795 908 1809"/>
994 <="" data-bbox="238 1809 908 1823"/>
995 <="" data-bbox="238 1823 908 1837"/>
996 <="" data-bbox="238 1837 908 1851"/>
997 <="" data-bbox="238 1851 908 1865"/>
998 <="" data-bbox="238 1865 908 1879"/>
999 <="" data-bbox="238 1879 908 1893"/>
1000 <="" data-bbox="238 1893 908 1907"/>
1001 <="" data-bbox="238 1907 908 1921"/>
1002 <="" data-bbox="238 1921 908 1935"/>
1003 <="" data-bbox="238 1935 908 1949"/>
1004 <="" data-bbox="238 1949 908 1963"/>
1005 <="" data-bbox="238 1963 908 1977"/>
1006 <="" data-bbox="238 1977 908 1991"/>
1007 <="" data-bbox="238 1991 908 2005"/>
1008 <="" data-bbox="238 2005 908 2019"/>
1009 <="" data-bbox="238 2019 908 2033"/>
1010 <="" data-bbox="238 2033 908 2047"/>
1011 <="" data-bbox="238 2047 908 2061"/>
1012 <="" data-bbox="238 2061 908 2075"/>
1013 <="" data-bbox="238 2075 908 2089"/>
1014 <="" data-bbox="238 2089 908 2103"/>
1015 <="" data-bbox="238 2103 908 2117"/>
1016 <="" data-bbox="238 2117 908 2131"/>
1017 <="" data-bbox="238 2131 908 2145"/>
1018 <="" data-bbox="238 2145 908 2159"/>
1019 <="" data-bbox="238 2159 908 2173"/>
1020 <="" data-bbox="238 2173 908 2187"/>
1021 <="" data-bbox="238 2187 908 2201"/>
1022 <="" data-bbox="238 2201 908 2215"/>
1023 <="" data-bbox="238 2215 908 2229"/>
1024 <="" data-bbox="238 2229 908 2243"/>
1025 <="" data-bbox="238 2243 908 2257"/>
1026 <="" data-bbox="238 2257 908 2271"/>
1027 <="" data-bbox="238 2271 908 2285"/>
1028 <="" data-bbox="238 2285 908 2299"/>
1029 <="" data-bbox="238 2299 908 2313"/>
1030 <="" data-bbox="238 2313 908 2327"/>
1031 <="" data-bbox="238 2327 908 2341"/>
1032 <="" data-bbox="238 2341 908 2355"/>
1033 <="" data-bbox="238 2355 908 2369"/>
1034 <="" data-bbox="238 2369 908 2383"/>
1035 <="" data-bbox="238 2383 908 2397"/>
1036 <="" data-bbox="238 2397 908 2411"/>
1037 <="" data-bbox="238 2411 908 2425"/>
1038 <="" data-bbox="238 2425 908 2439"/>
1039 <="" data-bbox="238 2439 908 2453"/>
1040 <="" data-bbox="238 2453 908 2467"/>
1041 <="" data-bbox="238 2467 908 2481"/>
1042 <="" data-bbox="238 2481 908 2495"/>
1043 <="" data-bbox="238 2495 908 2509"/>
1044 <="" data-bbox="238 2509 908 2523"/>
1045 <="" data-bbox="238 2523 908 2537"/>
1046 <="" data-bbox="238 2537 908 2551"/>
1047 <="" data-bbox="238 2551 908 2565"/>
1048 <="" data-bbox="238 2565 908 2579"/>
1049 <="" data-bbox="238 2579 908 2593"/>
1050 <="" data-bbox="238 2593 908 2607"/>
1051 <="" data-bbox="238 2607 908 2621"/>
1052 <="" data-bbox="238 2621 908 2635"/>
1053 <="" data-bbox="238 2635 908 2649"/>
1054 <="" data-bbox="238 2649 908 2663"/>
1055 <="" data-bbox="238 2663 908 2677"/>
1056 <="" data-bbox="238 2677 908 2691"/>
1057 <="" data-bbox="238 2691 908 2705"/>
1058 <="" data-bbox="238 2705 908 2719"/>
1059 <="" data-bbox="238 2719 908 2733"/>
1060 <="" data-bbox="238 2733 908 2747"/>
1061 <="" data-bbox="238 2747 908 2761"/>
1062 <="" data-bbox="238 2761 908 2775"/>
1063 <="" data-bbox="238 2775 908 2789"/>
1064 <="" data-bbox="238 2789 908 2803"/>
1065 <="" data-bbox="238 2803 908 2817"/>
1066 <="" data-bbox="238 2817 908 2831"/>
1067 <="" data-bbox="238 2831 908 2845"/>
1068 <="" data-bbox="238 2845 908 2859"/>
1069 <="" data-bbox="238 2859 908 2873"/>
1070 <="" data-bbox="238 2873 908 2887"/>
1071 <="" data-bbox="238 2887 908 2901"/>
1072 <="" data-bbox="238 2901 908 2915"/>
1073 <="" data-bbox="238 2915 908 2929"/>
1074 <="" data-bbox="238 2929 908 2943"/>
1075 <="" data-bbox="238 2943 908 2957"/>
1076 <="" data-bbox="238 2957 908 2971"/>
1077 <="" data-bbox="238 2971 908 2985"/>
1078 <="" data-bbox="238 2985 908 2999"/>
1079 <="" data-bbox="238 2999 908 3013"/>
1080 <="" data-bbox="238 3013 908 3027"/>
1081 <="" data-bbox="238 3027 908 3041"/>
1082 <="" data-bbox="238 3041 908 3055"/>
1083 <="" data-bbox="238 3055 908 3069"/>
1084 <="" data-bbox="238 3069 908 3083"/>
1085 <="" data-bbox="238 3083 908 3097"/>
1086 <="" data-bbox="238 3097 908 3111"/>
1087 <="" data-bbox="238 3111 908 3125"/>
1088 <="" data-bbox="238 3125 908 3139"/>
1089 <="" data-bbox="238 3139 908 3153"/>
1090 <="" data-bbox="238 3153 908 3167"/>
1091 <="" data-bbox="238 3167 908 3181"/>
1092 <="" data-bbox="238 3181 908 3195"/>
1093 <="" data-bbox="238 3195 908 3209"/>
1094 <="" data-bbox="238 3209 908 3223"/>
1095 <="" data-bbox="238 3223 908 3237"/>
1096 <="" data-bbox="238 3237 908 3251"/>
1097 <="" data-bbox="238 3251 908 3265"/>
1098 <="" data-bbox="238 3265 908 3279"/>
1099 <="" data-bbox="238 3279 908 3301"/>
1100 <="" data-bbox="238 3301 908 3323"/>
1101 <="" data-bbox="238 3323 908 3345"/>
1102 <="" data-bbox="238 3345 908 3367"/>
1103 <="" data-bbox="238 3367 908 3389"/>
1104 <="" data-bbox="238 3389 908 3411"/>
1105 <="" data-bbox="238 3411 908 3433"/>
1106 <="" data-bbox="238 3433 908 3455"/>
1107 <="" data-bbox="238 3455 908 3477"/>
1108 <="" data-bbox="238 3477 908 3499"/>
1109 <="" data-bbox="238 3499 908 3521"/>
1110 <="" data-bbox="238 3521 908 3543"/>
1111 <="" data-bbox="238 3543 908 3565"/>
1112 <="" data-bbox="238 3565 908 3587"/>
1113 <="" data-bbox="238 3587 908 3609"/>
1114 <="" data-bbox="238 3609 908 3631"/>
1115 <="" data-bbox="238 3631 908 3653"/>
1116 <="" data-bbox="238 3653 908 3675"/>
1117 <="" data-bbox="238 3675 908 3697"/>
1118 <="" data-bbox="238 3697 908 3719"/>
1119 <="" data-bbox="238 3719 908 3741"/>
1120 <="" data-bbox="238 3741 908 3763"/>
1121 <="" data-bbox="238 3763 908 3785"/>
1122 <="" data-bbox="238 3785 908 3807"/>
1123 <="" data-bbox="238 3807 908 3829"/>
1124 <="" data-bbox="238 3829 908 3851"/>
1125 <="" data-bbox="238 3851 908 3873"/>
1126 <="" data-bbox="238 3873 908 3895"/>
1127 <="" data-bbox="238 3895 908 3917"/>
1128 <="" data-bbox="238 3917 908 3939"/>
1129 <="" data-bbox="238 3939 908 3961"/>
1130 <="" data-bbox="238 3961 908 3983"/>
1131 <="" data-bbox="238 3983 908 4005"/>
1132 <="" data-bbox="238 4005 908 4027"/>
1133 <="" data-bbox="238 4027 908 4049"/>
1134 <="" data-bbox="238 4049 908 4071"/>
1135 <="" data-bbox="238 4071 908 4093"/>
1136 <="" data-bbox="238 4093 908 4115"/>
1137 <="" data-bbox="238 4115 908 4137"/>
1138 <="" data-bbox="238 4137 908 4159"/>
1139 <="" data-bbox="238 4159 908 4181"/>
1140 <="" data-bbox="238 4181 908 4203"/>
1141 <="" data-bbox="238 4203 908 4225"/>
1142 <="" data-bbox="238 4225 908 4247"/>
1143 <="" data-bbox="238 4247 908 4269"/>
1144 <="" data-bbox="238 4269 908 4291"/>
1145 <="" data-bbox="238 4291 908 4313"/>
1146 <="" data-bbox="238 4313 908 4335"/>
1147 <="" data-bbox="238 4335 908 4357"/>
1148 <="" data-bbox="238 4357 908 4379"/>
1149 <="" data-bbox="238 4379 908 4391"/>
1150 <="" data-bbox="238 4391 908 4413"/>
1151 <="" data-bbox="238 4413 908 4435"/>
1152 <="" data-bbox="238 4435 908 4457"/>
1153 <="" data-bbox="238 4457 908 4479"/>
1154 <="" data-bbox="238 4479 908 4491"/>
1155 <="" data-bbox="238 4491 908 4513"/>
1156 <="" data-bbox="238 4513 908 4535"/>
1157 <="" data-bbox="238 4535 908 4557"/>
1158 <="" data-bbox="238 4557 908 4579"/>
1159 <="" data-bbox="238 4579 908 4591"/>
1160 <="" data-bbox="238 4591 908 4613"/>
1161 <="" data-bbox="238 4613 908 4635"/>
1162 <="" data-bbox="238 4635 908 4657"/>
1163 <="" data-bbox="238 4657 908 4679"/>
1164 <="" data-bbox="238 4679 908 4691"/>
1165 <="" data-bbox="238 4691 908 4713"/>
1166 <="" data-bbox="238 4713 908 4735"/>
1167 <="" data-bbox="238 4735 908 4757"/>
1168 <="" data-bbox="238 4757 908 4779"/>
1169 <="" data-bbox="238 4779 908 4791"/>
1170 <="" data-bbox="238 4791 908 4813"/>
1171 <="" data-bbox="238 4813 908 4835"/>
1172 <="" data-bbox="238 4835 908 4857"/>
1173 <="" data-bbox="238 4857 908 4879"/>
1174 <="" data-bbox="238 4879 908 4891"/>
1175 <="" data-bbox="238 4891 908 4913"/>
1176 <="" data-bbox="238 4913 908 4935"/>
1177 <="" data-bbox="238 4935 908 4957"/>
1178 <="" data-bbox="238 4957 908 4979"/>
1179 <="" data-bbox="238 4979 908 4991"/>
1180 <="" data-bbox="238 4991 908 5013"/>
1181 <="" data-bbox="238 5013 908 5035"/>
1182 <="" data-bbox="238 5035 908 5057"/>
1183 <="" data-bbox="238 5057 908 5079"/>
1184 <="" data-bbox="238 5079 908 5091"/>
1185 <="" data-bbox="238 5091 908 5113"/>
1186 <="" data-bbox="238 5113 908 5135"/>
1187 <="" data-bbox="238 5135 908 5157"/>
1188 <="" data-bbox="238 5157 908 5179"/>
1189 <="" data-bbox="238 5179 908 5191"/>
1190 <="" data-bbox="238 5191 908 5213"/>
1191 <="" data-bbox="238 5213 908 5235"/>
1192 <="" data-bbox="238 5235 908 5257"/>
1193 <="" data-bbox="238 5257 908 5279"/>
1194 <="" data-bbox="238 5279 908 5291"/>
1195 <="" data-bbox="238 5291 908 5313"/>
1196 <="" data-bbox="238 5313 908 5335"/>
1197 <="" data-bbox="238 5335 908 5357"/>
1

```
        if len(priority_layers) > 0:
            implementation_model[PRIORITY_LAYERS_SEGMENT] = priority_la

        implementation_model_str = json.dumps(implementation_model)

        implementation_model_uuid = implementation_model[UUID_ATTRIBUTE]
        implementation_model_root = self._get_implementation_model_settings_bas

        with qgis_settings(implementation_model_root) as settings:
            settings.setValue(implementation_model_uuid, implementation_model_s
```

save_ncs_pathway

Code:

```
save_ncs_pathway(ncs_pathway)
```

Saves an NCS pathway object serialized to a json string indexed by the UUID.

Parameters:

Name	Type	Description	Default
ncs_pathway	Union[NcsPathway, dict]	NCS pathway object or attribute values required in a dictionary which are then serialized to a JSON string.	

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
705
706      Code:
707
708
709
710
711
712     def save_ncs_pathway(self, ncs_pathway: typing.Union[NcsPathway,
713         """Saves an NCS pathway object serialized to a json string
714         indexed by the UUID.
715
716
717         :param ncs_pathway: NCS pathway object or attribute values
718         in a dictionary which are then serialized to a JSON string.
719         :type ncs_pathway: NcsPathway, dict
720         """
721
722         if isinstance(ncs_pathway, NcsPathway):
723             ncs_pathway = ncs_pathway_to_dict(ncs_pathway)
724
725             ncs_str = json.dumps(ncs_pathway)
726
727             ncs_uuid = ncs_pathway[UUID_ATTRIBUTE]
728             ncs_root = self._get_ncs_pathway_settings_base()
729
730             with qgis_settings(ncs_root) as settings:
731                 settings.setValue(ncs_uuid, ncs_str)
```

save_priority_group

 **Code:**

```
save_priority_group(priority_group)
```

Save the priority group into the plugin settings

Parameters:

Name	Type	Description	Default
priority_group	str	Priority group	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
662
663
664  Code:
665
666
667
668
669
670     def save_priority_group(self, priority_group):
671         """Save the priority group into the plugin settings
672
673         :param priority_group: Priority group
674         :type priority_group: str
675
676
677         settings_key = self._get_priority_groups_settings_base(priority_group)
678
679         with qgis_settings(settings_key) as settings:
680             settings.setValue("name", priority_group["name"])
681             settings.setValue("value", priority_group["value"])
682             settings.setValue("description", priority_group.get("desc", ""))
```

save_priority_layer

Code :

```
save_priority_layer(priority_layer)
```

Save the priority layer into the plugin settings. Updates the layer with new priority groups.

Note: Emits priority_layers_changed signal

Parameters:

Name	Type	Description	Default
priority layer	dict	Priority layer required	

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

510
511
512  **Code:**
513
514
515
516
517 def save_priority_layer(self, priority_layer):
518 """Save the priority layer into the plugin settings.
519 Updates the layer with new priority groups.
520
521
522
523 Note: Emits priority_layers_changed signal
524
525
526
527 :param priority_layer: Priority layer
528 :type priority_layer: dict
529 """
530
531 settings_key = self._get_priority_layers_settings_base(priority_layer)
532
533
534 with qgis_settings(settings_key) as settings:
535 groups = priority_layer.get("groups", [])
536 settings.setValue("name", priority_layer["name"])
537 settings.setValue("description", priority_layer["description"])
538 settings.setValue("path", priority_layer["path"])
539 settings.setValue("selected", priority_layer.get("selected"))
540 settings.setValue("user_defined", priority_layer.get("user_defined"))
541 groups_key = f"{settings_key}/groups"
542 with qgis_settings(groups_key) as groups_settings:
543 for group_id in groups_settings.childGroups():
544 groups_settings.remove(group_id)
545 for group in groups:
546 group_key = f"{groups_key}/{group['name']}"
547 with qgis_settings(group_key) as group_settings:
548 group_settings.setValue("uuid", group.get("uuid"))
549 group_settings.setValue("name", group["name"])
550 group_settings.setValue("value", group["value"])
551
552
553 self.priority_layers_changed.emit()

```
save_scenario

Code:

save_scenario(scenario_settings)

Save the passed scenario settings into the plugin settings

Parameters:

| Name              | Type             | Description                | Default |
|-------------------|------------------|----------------------------|---------|
| scenario_settings | ScenarioSettings | Scenario settings required |         |



- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py


257
258
259
260
261
262
263
264
265
266
267
268
269
270

Code:

def save_scenario(self, scenario_settings):
    """Save the passed scenario settings into the plugin settings

    :param scenario_settings: Scenario settings
    :type scenario_settings: ScenarioSettings
    """

    settings_key = self._get_scenario_settings_base(scenario_setting

    self.save_scenario_extent(settings_key, scenario_settings.extent

    with qgis_settings(settings_key) as settings:
        settings.setValue("name", scenario_settings.name)
        settings.setValue("description", scenario_settings.description)
        settings.setValue("uuid", scenario_settings.uuid)
```

save_scenario_extent

 **Code:**

```
save_scenario_extent(key, extent)
```

Saves the scenario extent into plugin settings using the provided settings group key.

Parameters:

Name Type

key SpatialExtent

extent str Args: extent (SpatialExtent): Scenario extent key (str): QgsSettings group

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
272
273      Code:
274
275
276
277
278
279     def save_scenario_extent(self, key, extent):
280         """Saves the scenario extent into plugin settings
281         using the provided settings group key.
282
283         :param key: Scenario extent
284         :type key: SpatialExtent
285
286         :param extent: QgsSettings group key
287         :type extent: str
288
289
290
291     Args:
292         extent (SpatialExtent): Scenario extent
293         key (str): QgsSettings group key
294         """
295
296         spatial_extent = extent.spatial.bbox
297
298
299         spatial_key = f"{key}/extent/spatial/"
300         with qgis_settings(spatial_key) as settings:
301             settings.setValue("bbox", spatial_extent)
```

set_current_priority_layer

 **Code:**

```
set_current_priority_layer(identifier)
```

Set current priority layer

Parameters:

Name	Type	Description	Default
identifier	str	Priority layer identifier required	

• Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
541
542
543      Code:
544
545
546
547
548     def set_current_priority_layer(self, identifier):
549         """Set current priority layer
550
551         :param identifier: Priority layer identifier
552         :type identifier: str
553
554         """
555
556         with qgis_settings(
557             f"{self.BASE_GROUP_NAME}"/ f"{self.PRIORITY_LAYERS_GROUP_"
558         ) as settings:
559             for priority_layer in settings.childGroups():
560                 settings_key = self._get_priority_layers_settings_base_
561                 with qgis_settings(settings_key) as layer_settings:
562                     layer_settings.setValue(
563                         "selected", str(priority_layer) == str(identifier)
564                     )
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
```

set_value

 **Code:**

```
set_value(name, value)
```

Adds a new setting key and value on the plugin specific settings.

Parameters:

Name	Type	Description	Default
------	------	-------------	---------

name	str	Name of setting key required	
------	-----	------------------------------	--

value	Any	Value of the setting required	
-------	-----	-------------------------------	--

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
176
177     🧑‍💻 Code:
178
179
180
181
182
183     def set_value(self, name: str, value):
184         """Adds a new setting key and value on the plugin specific settings.
185
186
187         :param name: Name of setting key
188         :type name: str
189
190
191         :param value: Value of the setting
192         :type value: Any
193         """
194
195         self.settings.setValue(f"{self.BASE_GROUP_NAME}/{name}", value)
196         if isinstance(name, Settings):
197             name = name.value
198
199
200         self.settings_updated.emit(name, value)
```

update_implementation_model**🧑‍💻 Code:**

```
update_implementation_model(implementation_model)
```

Updates the attributes of the Implementation object in settings. On the path, the BASE_DIR in settings is used to reflect the absolute path of each NCS pathway layer. If BASE_DIR is empty then the NCS pathway setting will not be updated.

Parameters:

Name	Type	Description	Default
implementation_model	ImplementationModel	ImplementationModel object to be updated.	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
981
982
983  Code:
984
985
986
987
988     def update_implementation_model(self, implementation_model: ImplementationMc
989         """Updates the attributes of the Implementation object
990         in settings. On the path, the BASE_DIR in settings
991         is used to reflect the absolute path of each NCS
992         pathway layer. If BASE_DIR is empty then the NCS
993         pathway setting will not be updated.
994
995
996
997
998
999     :param implementation_model: ImplementationModel object to be updated.
1000     :type implementation_model: ImplementationModel
1001     """
1002
1003
1004     base_dir = self.get_value(Settings.BASE_DIR)
1005
1006
1007     if base_dir:
1008         # PWLs path update
1009         for layer in implementation_model.priority_layers:
1010             if layer in PRIORITY_LAYERS and base_dir not in layer.get(
1011                 PATH_ATTRIBUTE
1012             ):
1013                 abs_pwl_path = (
1014                     f"{base_dir}/{PRIORITY_LAYERS_SEGMENT}/"
1015                     f"{layer.get(PATH_ATTRIBUTE)}"
1016                 )
1017                 abs_pwl_path = str(os.path.normpath(abs_pwl_path))
1018                 layer[PATH_ATTRIBUTE] = abs_pwl_path
1019
1020
1021         # Remove then re-insert
1022         self.remove_implementation_model(str(implementation_model.uuid))
1023         self.save_implementation_model(implementation_model)
```

update_implementation_models

Code:

```
update_implementation_models()
```

Updates the attributes of the available implementation models

Parameters:

Name	Type	Description	Default
implementation_model	ImplementationModel	Implementation model object to be updated.	required

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py](#)

```
1010
1011
1012
1013
1014
1015
1016
1017 def update_implementation_models(self):
1018     """Updates the attributes of the available implementation models
1019
1020     :param implementation_model: Implementation model object to be updated.
1021     :type implementation_model: ImplementationModel
1022     """
1023
1024     models = self.get_all_implementation_models()
1025
1026     for implementation_model in models:
1027         self.update_implementation_model(implementation_model)
```

update_ncs_pathway**Code:**

```
update_ncs_pathway(ncs_pathway)
```

Updates the attributes of the NCS pathway object in settings. On the path, the BASE_DIR in settings is used to reflect the absolute path of each NCS pathway layer. If BASE_DIR is empty then the NCS pathway setting will not be updated, this only applies for default pathways.

Parameters:

Name	Type	Description	Default
ncs_pathway	NcsPathway	NCS pathway object to be updated.required	

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

Code :

```
805 def update_ncs_pathway(self, ncs_pathway):
806     """Updates the attributes of the NCS pathway object
807     in settings. On the path, the BASE_DIR in settings
808     is used to reflect the absolute path of each NCS
809     pathway layer. If BASE_DIR is empty then the NCS
810     pathway setting will not be updated, this only applies
811     for default pathways.
812
813     :param ncs_pathway: NCS pathway object to be updated.
814     :type ncs_pathway: NcsPathway
815     """
816
817     base_dir = self.get_value(Settings.BASE_DIR)
818     if not base_dir:
819         return
820
821
822     # Pathway location for default pathway
823     if not ncs_pathway.user_defined:
824         p = Path(ncs_pathway.path)
825         # Only update if path does not exist otherwise
826         # fallback to check under base directory.
827         if not p.exists():
828             abs_path = f"{base_dir}/{NCS_PATHWAY_SEGMENT}/" f"{p.name}"
829             abs_path = str(os.path.normpath(abs_path))
830             ncs_pathway.path = abs_path
831
832
833         # Carbon location
834         abs_carbon_paths = []
835         for cb_path in ncs_pathway.carbon_paths:
836             cp = Path(cb_path)
837             # Similarly, if the given carbon path does not exist then try
838             # to use the default one in the ncs_carbon directory.
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
999
```

```
if not cp.exists():
    abs_carbon_path = f"{base_dir}/{NCS_CARBON_SEGMENT}"/ f"{cp}
    abs_carbon_path = str(os.path.normpath(abs_carbon_path))
    abs_carbon_paths.append(abs_carbon_path)
else:
    abs_carbon_paths.append(cb_path)

ncs_pathway.carbon_paths = abs_carbon_paths

# Remove then re-insert
self.remove_ncs_pathway(str(ncs_pathway.uuid))
self.save_ncs_pathway(ncs_pathway)
```

update_ncs_pathways

Code:

```
update_ncs_pathways()
```

Updates the path attribute of all NCS pathway settings based on the BASE_DIR settings to reflect the absolute path of each NCS pathway layer. If BASE_DIR is empty then the NCS pathway settings will not be updated.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
787
788  Code:
789
790
791
792
793
794
795     def update_ncs_pathways(self):
796         """Updates the path attribute of all NCS pathway settings
797         based on the BASE_DIR settings to reflect the absolute path
798         of each NCS pathway layer.
799         If BASE_DIR is empty then the NCS pathway settings will not
800         be updated.
801         """
802
803         ncs_pathways = self.get_all_ncs_pathways()
804         for ncs in ncs_pathways:
805             self.update_ncs_pathway(ncs)
```

qgis_settings

Code:

```
qgis_settings(group_root, settings=None)
```

Context manager to help defining groups when creating QgsSettings.

:yields: Instance of the created settings :type: QgsSettings

Parameters:

Name	Type	Description	Default
group_root	str	Name of the root group for the settings required	
settings	QgsSettings	QGIS settings to use	None

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/conf.py

```
47
48  Code:
49
50
51
52
53
54 @contextlib.contextmanager
55
56 def qgis_settings(group_root: str, settings=None):
57     """Context manager to help defining groups when creating QgsSettings.
58
59     :param group_root: Name of the root group for the settings
60     :type group_root: str
61
62     :param settings: QGIS settings to use
63     :type settings: QgsSettings
64
65     :yields: Instance of the created settings
66     :rtype: QgsSettings
67
68     """
69
70     if settings is None:
71         settings = QgsSettings()
72     settings.beginGroup(group_root)
73     try:
74         yield settings
75     finally:
76         settings.endGroup()
```

Settings

Plugin global settings.

Covers the plugin global settings which a user can set and save. The settings will be saved using QgsSettings. Settings can be accessed via the QGIS options, a button on the docking widget, and from the toolbar menu.

CplusOptionsFactory

 **Code:**

```
CplusOptionsFactory()
```

Bases: **QgsOptionsWidgetFactory**

Options factory initializes the CPLUS settings.

Class which creates the widget required for the CPLUS settings. QgsOptionsWidgetFactory is used to accomplish this.

QGIS CPLUS Plugin Settings factory.

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/settings.py](#)

416
417
418
419
420

```
def __init__(self) -> None:  
    """QGIS CPLUS Plugin Settings factory."""  
    super().__init__()  
  
    self.setWindowTitle(OPTIONS_TITLE)
```

createWidget

 **Code:**

```
createWidget(parent)
```

Creates a widget for CPLUS settings.

Parameters:

Name	Type	Description	Default
parent	QWidget	Parent widget	required

Returns:

Type	Description
CplusSettings	Widget to be used in the QGIS options

• Source code in [/home/samwel/Workspace/Projects/cplus-plugin/src/cplus_plugin/settings.py](#)

431
432
433
434
435
436
437
438
439
440
441

 **Code:**

```
def createWidget(self, parent: QWidget) -> CplusSettings:
    """Creates a widget for CPLUS settings.

    :param parent: Parent widget
    :type parent: QWidget

    :returns: Widget to be used in the QGIS options
    :rtype: CplusSettings
    """

    return CplusSettings(parent)
```

 icon**Code:**`icon()`

Returns the icon which will be used for the CPLUS options tab.

Returns:**Type Description**

`QIcon` An icon object which contains the provided custom icon

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/settings.py`

```
422  
423  
424  
425  
426  
427  
428  
429
```

 **Code:**

```
def icon(self) -> QIcon:  
    """Returns the icon which will be used for the CPLUS options tab.  
  
    :returns: An icon object which contains the provided custom icon  
    :rtype: QIcon  
    """  
  
    return QIcon(ICONS_PATH)
```

CplusSettings **Code:**`CplusSettings(parent=None)`

Bases: `Ui_DlgSettings`, `QgsOptionsPageWidget`

QGIS CPLUS Plugin Settings dialog.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/settings.py`

 **Code:**

```
54
55
56
57
58
59
60
61 def __init__(self, parent=None) -> None:
62     """QGIS CPLUS Plugin Settings dialog."""
63     QgsOptionsPageWidget.__init__(self, parent)
64
65     self.setupUi(self)
66     self.message_bar = qgis.gui.QgsMessageBar(self)
67     self.layout().insertWidget(0, self.message_bar)
68
69     self.settings = qgis.core.QgsSettings()
70     settings_manager.settings_updated[str, object].connect(self.on_settings_change)
71
72     # Connections
73     self.cb_custom_logo.stateChanged.connect(self.logo_state_changed)
74     self.logo_file.fileChanged.connect(self.logo_file_changed)
75     self.folder_data.fileChanged.connect(self.base_dir_exists)
76
77     self.map_layer_file_widget.setStorageMode(QgsFileWidget.StorageMode.GetFile)
78     self.map_layer_box.layerChanged.connect(self.map_layer_changed)
79
80     self.resample_method_box.addItem(
81         tr("Nearest Neighbour"), QgsAlignRaster.ResampleAlg.RA_NearestNeighbour
82     )
83     self.resample_method_box.addItem(
84         tr("Bilinear (2x2 Kernel)"), QgsAlignRaster.ResampleAlg.RA_Bilinear
85     )
86     self.resample_method_box.addItem(
87         tr("Cubic (4x4 Kernel)"), QgsAlignRaster.ResampleAlg.RA_Cubic
88     )
89     self.resample_method_box.addItem(
90         tr("Cubic B-Spline (4x4 Kernel)"), QgsAlignRaster.ResampleAlg.RA_Cubics
91     )
```

```
    self.resample_method_box.addItem(
        tr("Lanczos (6x6 Kernel)'), QgsAlignRaster.ResampleAlg.RA_Lanczos
    )
    self.resample_method_box.addItem(
        tr("Average"), QgsAlignRaster.ResampleAlg.RA_Average
    )
    self.resample_method_box.addItem(tr("Mode"), QgsAlignRaster.ResampleAlg.RA_Mode)
    self.resample_method_box.addItem(
        tr("Maximum"), QgsAlignRaster.ResampleAlg.RA_Max
    )
    self.resample_method_box.addItem(
        tr("Minimum"), QgsAlignRaster.ResampleAlg.RA_Min
    )
    self.resample_method_box.addItem(
        tr("Median"), QgsAlignRaster.ResampleAlg.RA_Median
    )
    self.resample_method_box.addItem(
        tr("First Quartile (Q1)'), QgsAlignRaster.ResampleAlg.RA_Q1
    )
    self.resample_method_box.addItem(
        tr("Third Quartile (Q3)'), QgsAlignRaster.ResampleAlg.RA_Q3
    )
```

message_bar `instance-attribute`

 **Code:**

```
message_bar = qgis.gui.QgsMessageBar(self)
```

CPLUS plugin settings class.

Class which manages the CPLUS settings. Initializes the UI, which can be accessed from the menu drop-down or the QGIS settings.

apply

 **Code:**

apply()

This is called on OK click in the QGIS options panel.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/settings.py

108
109
110
111 **Code:**

```
def apply(self) -> None:  
    """This is called on OK click in the QGIS options panel."""  
  
    self.save_settings()
```

base_dir_exists

 **Code:**

base_dir_exists()

Checks if the provided base directory exists. A warning messages is presented if the directory does not exist.

Returns:**Type Description****bool** Whether the base directory exists

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/settings.py

```
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232
```

 **Code:**

```
def base_dir_exists(self) -> bool:  
    """Checks if the provided base directory exists.  
    A warning messages is presented if the directory does not exist.  
  
    :returns: Whether the base directory exists  
    :rtype: bool  
    """  
  
    # Clears the error messages when doing next check  
    self.message_bar.clearWidgets()  
  
    folder_found = False  
    base_dir_path = self.folder_data.filePath()  
    if not os.path.exists(base_dir_path):  
        # File not found  
        self.message_bar.pushWarning(  
            "CPLUS - Base directory not found: ", base_dir_path  
        )  
    else:  
        folder_found = True  
  
    return folder_found
```

closeEvent

 **Code:**`closeEvent(event)`

When closing the settings.

Parameters:

Name	Type	Description	Default
event	QShowEvent	Event that has been triggered required	
• Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/settings.py			

```
399
400
401
402
403
404
405
406
def closeEvent(self, event: QShowEvent) -> None:
    """When closing the settings.

        :param event: Event that has been triggered
        :type event: QShowEvent
    """

    super().closeEvent(event)
```

load_settings

 **Code:**`load_settings()`

Loads the settings and displays it in the options UI

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/settings.py`

312
313  **Code:**
314
315
316
317
318
319 def load_settings(self) -> None:
320 """Loads the settings and displays it in the options UI"""
321
322
323
324 # Analysis configuration settings
325
326
327 # Report settings
328 organization = settings_manager.get_value(
329 Settings.REPORT_ORGANIZATION, default="")
330)
331 self.txt_organization.setText(organization)
332
333
334
335
336 email = settings_manager.get_value(Settings.REPORT_CONTACT_EMAIL, default="")
337 self.txt_email.setText(email)
338
339
340
341 website = settings_manager.get_value(Settings.REPORT_WEBSITE, default="")
342 self.txt_website.setText(website)
343
344
345
346 custom_logo = int(
347 settings_manager.get_value(
348 Settings.REPORT_CUSTOM_LOGO,
349 default=True,
350)
351)
352 self.cb_custom_logo.setCheckState(custom_logo)
353 self.logo_file.setEnabled(custom_logo)
354
355
356
357
358
359
360 custom_logo_dir = settings_manager.get_value(
361 Settings.REPORT_LOGO_DIR, default=DEFAULT_LOGO_PATH
362)
363 self.logo_file.setFilePath(custom_logo_dir)
364 self.update_logo(custom_logo, custom_logo_dir)
365
366
367
368

```
369
370     footer = settings_manager.get_value(Settings.REPORT_FOOTER, default="")
371     self.txt_footer.setPlainText(footer)
372
373
374
375     disclaimer = settings_manager.get_value(Settings.REPORT_DISCLAIMER, def
376     self.txt_disclaimer.setPlainText(disclaimer)
377
378
379
380     report_license = settings_manager.get_value(Settings.REPORT_LICENSE, de
381     self.txt_license.setText(report_license)
382
383
384     # Advanced settings
385     base_dir = settings_manager.get_value(Settings.BASE_DIR, default="")
386     self.folder_data.setFilePath(base_dir)
387     self.base_dir_exists()
388
389
390     # Carbon layers coefficient
391     coefficient = settings_manager.get_value(
392         Settings.CARBON_COEFFICIENT, default=0.0
393     )
394     self.carbon_coefficient_box.setValue(float(coefficient))
395
396
397     # Pathway suitability index
398     pathway_suitability_index = settings_manager.get_value(
399         Settings.PATHWAY_SUITABILITY_INDEX, default=0
400     )
401     self.suitability_index_box.setValue(float(pathway_suitability_index))
402
403
404     # Snapping settings
405     self.snap_group_box.setChecked(
406         settings_manager.get_value(
407             Settings.SNAPPING_ENABLED, default=False, setting_type=bool
408         )
409     )
410     snap_layer_path = settings_manager.get_value(Settings.SNAP_LAYER, defau
411     self.map_layer_file_widget.setFilePath(snap_layer_path)
```

```
        self.rescale_values.setChecked(
            settings_manager.get_value(
                Settings.RESCALE_VALUES, default=False, setting_type=bool
            )
        )
        self.resample_method_box.setCurrentIndex(
            int(settings_manager.get_value(Settings.RESAMPLING_METHOD, default=0))
    )
```

logo_file_changed

Code:

```
logo_file_changed()
```

Called when the logo file directory changes. Will update the logo preview.

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/settings.py](#)

```
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
887
888
889
889
890
891
892
893
894
895
896
897
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
917
918
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1087
1088
1089
1089
1090
1091
1092
1093
1094
1095
1096
1096
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1188
1189
1190
1191
1192
1193
1194
1195
1196
1196
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1297
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1497
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1597
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1697
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2198
2199
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2297
2298
2299
2299
2300
2301
2302
2303
2304
2305
2306

```

logo_file_exists**Code:**`logo_file_exists()`

Checks if the provided logo directory exists. A warning messages is presented if the file cannot be found.

Returns:**Type Description**`bool` Whether the logo file exists

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/settings.py

188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209

 **Code:**

```
def logo_file_exists(self) -> bool:  
    """Checks if the provided logo directory exists.  
    A warning messages is presented if the file cannot be found.  
  
    :returns: Whether the logo file exists  
    :rtype: bool  
    """  
  
    # Clears the error messages when doing next check  
    self.message_bar.clearWidgets()  
  
    file_found = False  
    custom_logo_path = self.logo_file.filePath()  
    if not os.path.exists(custom_logo_path):  
        # File not found  
        self.message_bar.pushWarning(  
            "CPLUS - Custom logo not found: ", custom_logo_path  
        )  
    else:  
        file_found = True  
  
    # File found  
    return file_found
```

logo_state_changed **Code:**`logo_state_changed()`

Called when the custom logo option is disabled or enabled. Will update the logo preview.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/settings.py`

```
164
165  Code:
166
167
168
169
170
171     def logo_state_changed(self) -> None:
172         """Called when the custom logo option is disabled or enabled.
173         Will update the logo preview.
174         """
175
176         custom_logo = self.cb_custom_logo.checkState()
177         custom_logo_path = self.logo_file.filePath()
178
179         # Enables/disables the file widget for the logo directory
180         if custom_logo:
181             self.logo_file.setEnabled(True)
182         else:
183             self.logo_file.setEnabled(False)
184
185         self.update_logo(custom_logo, custom_logo_path)
```

map_layer_changed**Code:**

```
map_layer_changed(layer)
```

Sets the file path of the selected layer in file path input

Parameters:

Name	Type	Description	Default
layer	QgsMapLayer	Qgis map layer required	

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/settings.py

```
113
114
115
116
117
118
119
120
def map_layer_changed(self, layer):
    """Sets the file path of the selected layer in file path input

    :param layer: Qgis map layer
    :type layer: QgsMapLayer
    """
    if layer is not None:
        self.map_layer_file_widget.setFilePath(layer.source())
```

on_settings_changed**Code:**

```
on_settings_changed(name, value)
```

Slot raised when settings has been changed.

Parameters:

Name	Type	Description	Default
name	str	Name of the setting that has changed. required	
value	Any	New value for the given settings name.required	

• Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/settings.py

```
122
123
124  Code:
125
126
127
128
129
130     def on_settings_changed(self, name: str, value: typing.Any):
131         """Slot raised when settings has been changed.
132
133         :param name: Name of the setting that has changed.
134         :type name: str
135
136         :param value: New value for the given settings name.
137         :type value: Any
138         """
139
140         # Create NCS pathway subdirectory if base directory has changed.
141         if name == Settings.BASE_DIR.value:
142             if not value:
143                 return
144
145             FileUtils.create_ncs_pathways_dir(value)
146             FileUtils.create_ncs_carbon_dir(value)
147             FileUtils.create_pwls_dir(value)
```

save_settings **Code:**`save_settings()`

Saves the settings. Also does error checking for settings (e.g if the custom logo exists). Will present the user with an error message if an issue is found.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/settings.py`

234
235  **Code:**

236
237
238
239
240
241 def save_settings(self) -> None:
242 """Saves the settings.
243 Also does error checking for settings (e.g if the custom logo exists).
244 Will present the user with an error message if an issue is found.
245 """
246
247
248
249
250 # Analysis configuration
251
252
253
254 # Report settings
255 organization = self.txt_organization.text()
256 settings_manager.set_value(Settings.REPORT_ORGANIZATION, organization)
257
258
259 email = self.txt_email.text()
260 settings_manager.set_value(Settings.REPORT_CONTACT_EMAIL, email)
261
262
263
264 website = self.txt_website.text()
265 settings_manager.set_value(Settings.REPORT_WEBSITE, website)
266
267
268
269 custom_logo = self.cb_custom_logo.checkState()
270 settings_manager.set_value(Settings.REPORT_CUSTOM_LOGO, custom_logo)
271
272
273
274 # Checks if the logo file exists if custom logo is enabled
275 if custom_logo:
276 custom_logo_path = self.logo_file.filePath()
277 settings_manager.set_value(Settings.REPORT_LOGO_DIR, custom_logo_path)
278
279
280
281
282 if not os.path.exists(custom_logo_path):
283 # File not found, disable custom logo
284 settings_manager.set_value(Settings.REPORT_CUSTOM_LOGO, False)
285
286
287
288 iface.messageBar().pushWarning(
289
290

```
291         "CPLUS - Custom logo not found, disabled: ", custom_logo_path
292     )
293
294
295     footer = self.txt_footer.toPlainText()
296     settings_manager.set_value(Settings.REPORT_FOOTER, footer)
297
298
299     disclaimer = self.txt_disclaimer.toPlainText()
300     settings_manager.set_value(Settings.REPORT_DISCLAIMER, disclaimer)
301
302
303     report_license = self.txt_license.text()
304     settings_manager.set_value(Settings.REPORT_LICENSE, report_license)
305
306
307     # Advanced settings
308     base_dir_path = self.folder_data.filePath()
309     settings_manager.set_value(Settings.BASE_DIR, base_dir_path)
310
311
312     # Carbon layers coefficient saving
313     coefficient = self.carbon_coefficient_box.value()
314     settings_manager.set_value(Settings.CARBON_COEFFICIENT, coefficient)
315
316
317     # Pathway suitability index
318     pathway_suitability_index = self.suitability_index_box.value()
319     settings_manager.set_value(
320         Settings.PATHWAY_SUITABILITY_INDEX, pathway_suitability_index
321     )
322
323
324     # Snapping settings saving
325
326     settings_manager.set_value(
327         Settings.SNAPPING_ENABLED, self.snap_group_box.isChecked()
328     )
329     snap_layer_path = self.map_layer_file_widget.filePath()
330     settings_manager.set_value(Settings.SNAP_LAYER, snap_layer_path)
331
332
333     settings_manager.set_value(
334         Settings.RESCALE_VALUES, self.rescale_values.isChecked()
```

```
)  
    settings_manager.set_value(  
        Settings.RESAMPLING_METHOD, self.resample_method_box.currentIndex()  
)  
  
    # Checks if the provided base directory exists  
    if not os.path.exists(base_dir_path):  
        iface.messageBar().pushCritical(  
            "CPLUS - Base directory not found: ", base_dir_path  
)
```

showEvent

Code:

```
showEvent(event)
```

Show event being called. This will display the plugin settings. The stored/saved settings will be loaded.

Parameters:

Name	Type	Description	Default
event	QShowEvent	Event that has been triggered	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/settings.py

```
388
389  Code:
390
391
392
393
394
395     def showEvent(self, event: QShowEvent) -> None:
396         """Show event being called. This will display the plugin settings.
397         The stored/saved settings will be loaded.
398
399         :param event: Event that has been triggered
400         :type event: QShowEvent
401         """
402
403
404         super().showEvent(event)
405         self.load_settings()
```

update_logo

```
 Code:
update_logo(custom_logo, logo_dir=DEFAULT_LOGO_PATH)
```

Updates the logo preview.

If the logo is not found, the default logo will be used.

Parameters:

Name	Type	Description	Default
custom_logo	bool	If a custom logo should be used	required
logo_dir	str	The custom logo directory	DEFAULT_LOGO_PATH

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/settings.py

140
141  **Code:**
142
143
144
145
146
147 def update_logo(self, custom_logo, logo_dir=DEFAULT_LOGO_PATH):
148 """Updates the logo preview.
149
150 If the logo is not found, the default logo will be used.
151
152 :param custom_logo: If a custom logo should be used
153 :type custom_logo: bool
154
155 :param logo_dir: The custom logo directory
156 :type logo_dir: str
157 """
158
159 logo_found = False
160 if custom_logo:
161 # If custom logo is active, check if the provided directory exists
162 logo_found = self.logo_file_exists()
163
164 if custom_logo and logo_found:
165 # If custom logo is enabled and the logo file exists
166 pixmap = QPixmap(logo_dir)
167 else:
168 # If custom logo is disabled. The default logo will also be used when t
169 pixmap = QPixmap(DEFAULT_LOGO_PATH)
170 self.lbl_logo_image.setPixmap(pixmap)

Utilities

Plugin utilities

FileUtils

Provides functionality for commonly used file-related operations.

`create_ncs_carbon_dir` | `staticmethod`

 **Code:**

```
create_ncs_carbon_dir(base_dir)
```

Creates an NCS subdirectory for carbon layers under BASE_DIR. Skips creation of the subdirectory if it already exists.

- Source code in `/home/samwelij/Workspace/Projects/cplus-plugin/src/cplus_plugin/utils.py`

```
401
402
403  Code:
404
405
406
407
408 @staticmethod
409
410 def create_ncs_carbon_dir(base_dir: str):
411     """Creates an NCS subdirectory for carbon layers under BASE_DIR.
412     Skips creation of the subdirectory if it already exists.
413     """
414
415     if not Path(base_dir).is_dir():
416         return
417
418     ncs_carbon_dir = f"{base_dir}/{NCS_CARBON_SEGMENT}"
419     message = tr("Missing parent directory when creating NCS carbon sub")
420     FileUtils.create_new_dir(ncs_carbon_dir, message)
```

create_ncs_pathways_dir | staticmethod

 **Code:**

```
create_ncs_pathways_dir(base_dir)
```

Creates an NCS subdirectory under BASE_DIR. Skips creation of the subdirectory if it already exists.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/utils.py

```
387
388     @staticmethod
389     def create_ncs_pathways_dir(base_dir: str):
390         """Creates an NCS subdirectory under BASE_DIR. Skips
391         creation of the subdirectory if it already exists.
392         """
393
394         if not Path(base_dir).is_dir():
395             return
396
397         ncs_pathway_dir = f"{base_dir}/{NCS_PATHWAY_SEGMENT}"
398         message = tr(
399             "Missing parent directory when creating NCS pathways subdirectory."
400         )
401         FileUtils.create_new_dir(ncs_pathway_dir, message)
```

create_new_dir staticmethod

 **Code:**

```
create_new_dir(directory, log_message='')
```

Creates new file directory if it doesn't exist

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/utils.py

426
427
428
429
430
431
432
433
434

```
@staticmethod  
def create_new_dir(directory: str, log_message: str = ""):  
    """Creates new file directory if it doesn't exist"""  
    p = Path(directory)  
    if not p.exists():  
        try:  
            p.mkdir()  
        except FileNotFoundError:  
            log(log_message)
```

create_new_file staticmethod

 **Code:**

```
create_new_file(file_path, log_message='')
```

Creates new file

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/utils.py

```
436
437  Code:
438
439
440
441
442
443
444
445 @staticmethod
def create_new_file(file_path: str, log_message: str = ""):
    """Creates new file"""
    p = Path(file_path)

    if not p.exists():
        try:
            p.touch(exist_ok=True)
        except FileNotFoundError:
            log(log_message)
```

create_pwls_dir

 **Code:**

```
create_pwls_dir(base_dir)
```

Creates priority weighting layers subdirectory under BASE_DIR. Skips creation of the subdirectory if it already exists.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/utils.py

```
413  
414  
415     Code:  
416  
417  
418  
419  
420  
421  
422  
423  
424
```

```
def create_pwls_dir(base_dir: str):  
    """Creates priority weighting layers subdirectory under BASE_DIR.  
    Skips creation of the subdirectory if it already exists.  
    """  
  
    if not Path(base_dir).is_dir():  
        return  
  
    pwl_dir = f"{base_dir}/{PRIORITY_LAYERS_SEGMENT}"  
    message = tr(  
        "Missing parent directory when creating priority weighting layers subdi-  
    )  
    FileUtils.create_new_dir(pwl_dir, message)
```

get_icon | staticmethod

```
Code:
```

```
get_icon(file_name)
```

Creates an icon based on the icon name in the 'icons' folder.

Parameters:

Name	Type	Description	Default
file_name	str	File name which should include the extension.	required

Returns:

Type	Description
QtGui.QIcon	Icon object matching the file name.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/utils.py

```
350
351  Code:
352
353
354
355
356
357 @staticmethod
358
359     def get_icon(file_name: str) -> QtGui.QIcon:
360         """Creates an icon based on the icon name in the 'icons' folder.
361
362         :param file_name: File name which should include the extension.
363         :type file_name: str
364
365         :returns: Icon object matching the file name.
366         :rtype: QtGui.QIcon
367         """
368
369         icon_path = os.path.normpath(f"{FileUtils.plugin_dir()}/icons/{file_name}")
370
371         if not os.path.exists(icon_path):
372             return QtGui.QIcon()
373
374         return QtGui.QIcon(icon_path)
```

plugin_dir | staticmethod

 **Code:**

```
plugin_dir()
```

Returns the root directory of the plugin.

Returns:

Type Description

str Root directory of the plugin.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/utils.py

```
341
342
343
344
345
346
347
348
    @staticmethod
    def plugin_dir() -> str:
        """Returns the root directory of the plugin.

        :returns: Root directory of the plugin.
        :rtype: str
        """
        return os.path.join(os.path.dirname(os.path.realpath(__file_
```

report_template_path | staticmethod

 **Code:**

```
report_template_path(file_name=None)
```

Get the absolute path to the template file with the given name. Caller needs to verify that the file actually exists.

Parameters:

Name	Type	Description	Default
file_name	str	Template file name including the extension. If none is specified then it will use main.qpt as the default template name.	None

Returns:

Type Description

str The absolute path to the template file with the given name.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/utils.py

```
367
368  Code:
369
370
371
372
373
374 @staticmethod
375
376 def report_template_path(file_name=None) -> str:
377     """Get the absolute path to the template file with the given name.
378     Caller needs to verify that the file actually exists.
379
380     :param file_name: Template file name including the extension. If
381     none is specified then it will use `main.qpt` as the default
382     template name.
383     :type file_name: str
384
385     :returns: The absolute path to the template file with the given name.
386     :rtype: str
387     """
388
389     if file_name is None:
390         file_name = TEMPLATE_NAME
391
392     absolute_path = f"{FileUtils.plugin_dir()}/data/reports/{file_name}"
393
394     return os.path.normpath(absolute_path)
```

Code: ``` align_rasters(input_raster_source, reference_raster_source, extent=None, output_dir=None, rescale_values=False, resample_method=0,) ```

Based from work on <https://github.com/inasafe/inasafe/pull/2070> Aligns the passed raster files source and save the results into new files.

Parameters:

Name	Type	Description	Default
input_raster_source	str	Input layer source	required
reference_raster_source	str	Reference layer source	required
extent	list	Clip extent	None
output_dir	str	Absolute path of the output directory for the snapped layers	None
rescale_values	bool	Whether to rescale pixel values	False
resample_method	QgsAlignRaster.ResampleAlg	Method to use when resampling	0

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/utils.py`

228
229
230  **Code:**
231
232
233
234
235
236 def align_rasters(
237 input_raster_source,
238 reference_raster_source,
239 extent=None,
240 output_dir=None,
241 rescale_values=False,
242 resample_method=0,
243):
244 """
245 Based from work on <https://github.com/inasafe/inasafe/pull/2070>
246 Aligns the passed raster files source and save the results into new files.
247
248 :param input_raster_source: Input layer source
249 :type input_raster_source: str
250
251 :param reference_raster_source: Reference layer source
252 :type reference_raster_source: str
253
254 :param extent: Clip extent
255 :type extent: list
256
257 :param output_dir: Absolute path of the output directory for the snapped
258 layers
259 :type output_dir: str
260
261 :param rescale_values: Whether to rescale pixel values
262 :type rescale_values: bool
263
264 :param resample_method: Method to use when resampling
265 :type resample_method: QgsAlignRaster.ResampleAlg
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284

```
285     """
286
287
288     try:
289         snap_directory = os.path.join(output_dir, "snap_layers")
290
291         FileUtils.create_new_dir(snap_directory)
292
293         input_path = Path(input_raster_source)
294
295         input_layer_output = os.path.join(
296             f"{snap_directory}", f"{input_path.stem}_{str(uuid.uuid4())[:4]}.tif"
297         )
298
299         FileUtils.create_new_file(input_layer_output)
300
301         align = QgsAlignRaster()
302         lst = [
303             QgsAlignRaster.Item(input_raster_source, input_layer_output),
304         ]
305
306         resample_method_value = QgsAlignRaster.ResampleAlg.RA_NearestNeighbour
307
308         try:
309             resample_method_value = QgsAlignRaster.ResampleAlg(int(resample_method_value))
310         except Exception as e:
311             log(f"Problem creating a resample value when snapping, {e}")
312
313         if rescale_values:
314             lst[0].rescaleValues = rescale_values
315
316         lst[0].resampleMethod = resample_method_value
317
318         align.setRasters(lst)
319         align.setParametersFromRaster(reference_raster_source)
320
321         layer = QgsRasterLayer(input_raster_source, "input_layer")
```

```
if extent:
    original_extent = QgsRectangle(
        float(extent[0]), float(extent[1]), float(extent[2]), float(extent[3]))
    source_crs = QgsCoordinateReferenceSystem("EPSG:4326")
else:
    original_extent = layer.extent()
    source_crs = QgsCoordinateReferenceSystem(layer.crs())

destination_crs = QgsCoordinateReferenceSystem(align.destinationCrs())

transform = QgsCoordinateTransform(
    source_crs, destination_crs, QgsProject.instance())
)

tranformed_extent = transform.transformBoundingBox(original_extent)

align.setClipExtent(tranformed_extent)

if not align.run():
    log(
        f"Problem during snapping for {input_raster_source} and {reference_layer}")
    raise Exception(align.errorMessage())
except Exception as e:
    log(
        f"Problem occured when snapping, {str(e)}."
        f" Update snap settings and re-run the analysis"
    )

return None, None

log(
    f"Finished snapping using resampling method {resample_method_value.name}"
    f" original layer - {input_raster_source}, "
    f"snapped output - {input_layer_output} \n"
```

```
)  
  
    return input_layer_output, None
```

calculate_raster_value_area

Code:

```
calculate_raster_value_area(  
    layer, band_number=1, feedback=None  
)
```

Calculates the area of value pixels for the given band in a raster layer.

Please note that this function will run in the main application thread hence for best results, it is recommended to execute it in a background process if part of a bigger workflow.

Parameters:

Name	Type	Description	Default
layer	QgsRasterLayer	Input layer whose area for value pixels is to required be calculated.	
band_number	int	Band number to compute area, default is band one.	1
feedback	QgsProcessingFeedback	Feedback object for progress during area calculation.	None

Returns:

Type Description

float A dictionary containing the pixel value as the key and the corresponding area in hectares as the value for all the pixels in the raster otherwise returns a empty dictionary if the raster is invalid or if it is empty.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/utils.py`

155
156  **Code:**
157
158
159
160
161
162 **def calculate_raster_value_area(**
163 **layer: QgsRasterLayer, band_number: int = 1, feedback: QgsProcessingFeedback**
164 **) -> dict:**
165 <="" data-bbox="238 298 998 315"/>
166 <="" data-bbox="238 333 998 398"/>
167 <="" data-bbox="238 416 998 451"/>
168 <="" data-bbox="238 469 998 504"/>
169 <="" data-bbox="238 522 998 557"/>
170 <="" data-bbox="238 575 998 640"/>
171 <="" data-bbox="238 658 998 693"/>
172 <="" data-bbox="238 711 998 746"/>
173 <="" data-bbox="238 764 998 809"/>
174 <="" data-bbox="238 827 998 862"/>
175
176 <="" data-bbox="238 416 998 451"/>
177 <="" data-bbox="238 469 998 504"/>
178 <="" data-bbox="238 522 998 557"/>
179 <="" data-bbox="238 575 998 610"/>
180 <="" data-bbox="238 628 998 663"/>
181 <="" data-bbox="238 681 998 716"/>
182 <="" data-bbox="238 734 998 769"/>
183 <="" data-bbox="238 787 998 822"/>
184
185
186 <="" data-bbox="238 522 998 557"/>
187 <="" data-bbox="238 575 998 610"/>
188 <="" data-bbox="238 628 998 663"/>
189
190 <="" data-bbox="238 586 998 621"/>
191 <="" data-bbox="238 639 998 674"/>
192 <="" data-bbox="238 692 998 727"/>
193 <="" data-bbox="238 745 998 780"/>
194 <="" data-bbox="238 798 998 833"/>
195 <="" data-bbox="238 851 998 886"/>
196
197 <="" data-bbox="238 664 998 709"/>
198 <="" data-bbox="238 727 998 762"/>
199
200 <="" data-bbox="238 746 998 781"/>
201 <="" data-bbox="238 799 998 834"/>
202 <="" data-bbox="238 852 998 887"/>
203
204
205
206 <="" data-bbox="238 777 998 812"/>
207 <="" data-bbox="238 830 998 865"/>
208 <="" data-bbox="238 873 998 908"/>
209
210 <="" data-bbox="238 886 998 921"/>
211

```
212     "BAND": band_number,
213     "OUTPUT_TABLE": "TEMPORARY_OUTPUT",
214     "OUTPUT_HTML_FILE": "[Skip output]",
215   }
216
217
218
219
220   algorithm_result = processing.run(algorithm_name, params, feedback=feedback)
221
222
223   # Get number of pixels with values
224   total_pixel_count = algorithm_result["TOTAL_PIXEL_COUNT"]
225   if total_pixel_count == 0:
226     log("Input layer for raster area calculation is empty.", info=False)
227     return {}
228
229
230   output_table = algorithm_result["OUTPUT_TABLE"]
231   if output_table is None:
232     log("Unique values raster table could not be retrieved.", info=False)
233     return {}
234
235
236   area_calc = QgsDistanceArea()
237   crs = layer.crs()
238   area_calc.setSourceCrs(crs, QgsCoordinateTransformContext())
239   if crs is not None:
240     # Use ellipsoid calculation if available
241     area_calc.setEllipsoid(crs.ellipsoidAcronym())
242
243
244   version = Qgis.versionInt()
245   if version < 33000:
246     unit_type = QgsUnitTypes.AreaUnit.AreaHectares
247   else:
248     unit_type = Qgis.AreaUnit.Hectares
249
250
251   pixel_areas = []
252   features = output_table.getFeatures()
253   for f in features:
254     pixel_value = f.attribute(0)
255     area = f.attribute(2)
```

```
pixel_value_area = area_calc.convertAreaMeasurement(area, unit_type)
pixel_areas[pixel_value] = pixel_value_area

return pixel_areas
```

clean_filename

Code:

```
clean_filename(filename)
```

Creates a safe filename by removing operating system invalid filename characters.

Parameters:

Name	Type	Description	Default
filename	str	File name	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/utils.py

```
136
137  Code:
138
139
140
141
142
143     def clean_filename(filename):
144         """Creates a safe filename by removing operating system
145         invalid filename characters.
146
147         :param filename: File name
148         :type filename: str
149
150         :returns A clean file name
151         :rtype str
152         """
153
154         characters = " %:/,\\[ ]<>*?"
155
156         for character in characters:
157             if character in filename:
158                 filename = filename.replace(character, "_")
159
160
161         return filename
```

get_plugin_version

 **Code:**

```
get_plugin_version()
```

Returns the current plugin version as saved in the metadata.txt plugin file.

Returns:

Type Description

str Plugin version

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/utils.py

```
96
97      Code:
98
99
100
101
102
103     def get_plugin_version() -> [str, None]:
104         """Returns the current plugin version
105         as saved in the metadata.txt plugin file.
106
107         :returns version: Plugin version
108         :rtype version: str
109         """
110
111         metadata_file = Path(__file__).parent.resolve() / "metadata.txt"
112
113         with open(metadata_file, "r") as f:
114             for line in f.readlines():
115                 if line.startswith("version"):
116                     version = line.strip().split("=")[1]
117                     return version
118
119         return None
```

get_report_font

 **Code:**

```
get_report_font(size=11.0, bold=False, italic=False)
```

Uses the default font family name to create a font for use in the report.

Parameters:

Name	Type	Description	Default
size	float	The font point size, default is 11.	11.0
bold	bool	True for bold font else False which is the default.	False
italic	bool	True for font to be in italics else False which is the default.	False

Returns:

Type	Description
QtGui.QFont	Font to use in a report.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/utils.py

```
113
114
115      Code:
116
117
118
119
120     def get_report_font(size=11.0, bold=False, italic=False) -> QtGui.QFont:
121         """Uses the default font family name to create a
122         font for use in the report.
123
124
125
126         :param size: The font point size, default is 11.
127         :type size: float
128
129
130         :param bold: True for bold font else False which is the default.
131         :type bold: bool
132
133
134         :param italic: True for font to be in italics else False which is the default.
135         :type italic: bool
136
137
138         :returns: Font to use in a report.
139         :rtype: QtGui.QFont
140
141         """
142
143         font_weight = 50
144         if bold is True:
145             font_weight = 75
146
147
148         return QtGui.QFont(REPORT_FONT_NAME, int(size), font_weight, italic)
```

log

 **Code:**

```
log(message, name='qgis_cplus', info=True, notify=True)
```

Logs the message into QGIS logs using qgis_cplus as the default log instance. If notify_user is True, user will be notified about the log.

Parameters:

Name	Type	Description	Default
message	str	The log message	required
name	str	Name of te log instance, qgis_cplus is the default	'qgis_cplus'
info	bool	Whether the message is about info or a warning	True
notify	bool	Whether to notify user about the log	True

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/utils.py`

52
53
54  **Code:**
55
56
57
58
59 def log(
60 message: str,
61 name: str = "qgis_cplus",
62 info: bool = True,
63 notify: bool = True,
64):
65 """Logs the message into QGIS logs using qgis_cplus as the default
66 log instance.
67 If notify_user is True, user will be notified about the log.
68
69 :param message: The log message
70 :type message: str
71
72 :param name: Name of te log instance, qgis_cplus is the default
73 :type message: str
74
75 :param info: Whether the message is about info or a
76 warning
77 :type info: bool
78
79 :param notify: Whether to notify user about the log
80 :type notify: bool
81 """
82 level = Qgis.Info if info else Qgis.Warning
83 QgsMessageLog.logMessage(
84 message,
85 name,
86 level=level,
87 notifyUser=notify,
88)
89

open_documentation **Code:**`open_documentation(url=None)`

Opens documentation website in the default browser

Parameters:

Name	Type	Description	Default
url	str	URL link to documentation site (e.g. gh pages site)	None

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/utils.py

```
84
85  Code:
86
87
88
89
90
91 def open_documentation(url=None):
92     """Opens documentation website in the default browser
93
94     :param url: URL link to documentation site (e.g. gh pages site)
95     :type url: str
96
97     """
98
99     url = DOCUMENTATION_SITE if url is None else url
100    result = QtGui.QDesktopServices.openUrl(QtCore.QUrl(url))
101    return result
```

tr

 **Code:**

tr(message)

Get the translation for a string using Qt translation API. We implement this ourselves since we do not inherit QObject.

Parameters:

Name	Type	Description	Default
message	str, QString	String for translation.required	

Returns:

Type	Description
QString	Translated version of message.

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/utils.py](#)

38
39
40
41
42
43
44
45
46
47
48
49 **Code:**

```
def tr(message):
    """Get the translation for a string using Qt translation API.
    We implement this ourselves since we do not inherit QObject.

    :param message: String for translation.
    :type message: str, QString

    :returns: Translated version of message.
    :rtype: QString
    """

    # noinspection PyTypeChecker,PyArgumentList,PyCallByClass
    return QtCore.QCoreApplication.translate("QgisCplus", message)
```

Reports

Report Generator

CPLUS Report generator.

ReportGenerator

 **Code:**

```
ReportGenerator(context, feedback=None)
```

Generator for CPLUS reports.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/generator.py

```
226
227  Code:
228
229
230
231
232
233     def __init__(self, context: ReportContext, feedback: QgsFeedback = None):
234         self._context = context
235         self._feedback = context.feedback or feedback
236         self._error_messages: typing.List[str] = []
237         self._layout = None
238         self._project = None
239         self._variable_register = LayoutVariableRegister()
240         self._report_output_dir = ""
241         self._output_layout_path = ""
242         self._repeat_page = None
243         self._repeat_page_num = -1
244         self._repeat_item = None
245         self._reference_layer_group = None
246         self._scenario_layer = None
247         self._area_processing_feedback = None
248         self._implementation_models_area = {}

249         if self._feedback:
250             self._feedback.canceled.connect(self._on_feedback_cancelled)

251             self._area_calculation_progress_reference = 0
252             self._area_calculation_step_increment = 0
```

`context` `property` **Code:**

context

Returns the report context used by the generator.

Returns:

Type	Description
------	-------------

ReportContext	Report context object used by the generator.
---------------	--

`feedback` `property` **Code:**

feedback

Returns the feedback object for process update and cancellation.

Returns:

Type	Description
------	-------------

QgsFeedback	Feedback object or None if not specified.
-------------	---

`layout` `property` **Code:**

layout

Returns the layout object used to generate the report.

Returns:

Type	Description
------	-------------

`QgsPrintLayout` The layout object used to generate the report or None if the process was not successful.

`output_dir` `property`

 **Code:**

`output_dir`

Creates, if it does not exist, the output directory where the analysis reports will be saved. This is relative to the base directory and scenario output sub-folder.

Returns:

Type	Description
------	-------------

`str` Output directory where the analysis reports will be saved.

`output_layout_path` `property`

 **Code:**

`output_layout_path`

Absolute path to a temporary file containing the layout as a QPT file.

When this object is used within a QgsTask, it is recommended to use this layout path to reconstruct the layout rather calling the `layout` attribute since it was created in a separate thread.

Returns:

Type	Description
------	-------------

`str` Path to the layout template file.

repeat_page `property`

 **Code:**

```
repeat_page
```

Returns the page item that will be repeated based on the number of implementation models in the scenario.

A repeat page is a layout page item that contains the first instance of a CplusMapRepeatItem.

Returns:

Type	Description
<code>QgsLayoutItemPage</code>	Page item containing a CplusMapRepeatItem or None if not found.

`duplicate_repeat_page`

 **Code:**

```
duplicate_repeat_page(position)
```

Duplicates the repeat page and adds it to the layout at the given position.

Parameters:

Name	Type	Description	Default
<code>position</code>	<code>int</code>	Zero-based position to insert the duplicated page. If the position is greater required than the number of pages, then the duplicated page will be inserted at the end of the layout.	

Returns:

Type	Description
<code>bool</code>	True if the page was successfully duplicated else False.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/generator.py

438
439
440  **Code:**
441
442
443
444
445 def duplicate_repeat_page(self, position: int) -> bool:
446 """Duplicates the repeat page and adds it to the layout
447 at the given position.
448
449 :param position: Zero-based position to insert the duplicated page. If
450 the position is greater than the number of pages, then the
451 duplicated page will be inserted at the end of the layout.
452 :type position: int
453
454 :returns: True if the page was successfully duplicated else False.
455 :rtype: bool
456 """
457
458 if self._repeat_page is None:
459 return False
460
461 if self._layout is None:
462 return False
463
464 if self._repeat_page_num == -1:
465 tr_msg = "Repeat page not found in page collection"
466 self._error_messages.append(tr_msg)
467 return False
468
469 new_page = QgsLayoutItemPage(self._layout)
470 new_page.attemptResize(self._repeat_page.sizeWithUnits())
471 new_page.setPageStyleSymbol(self._repeat_page.pageStyleSymbol().clone())
472
473 # Insert empty repeat page at the given position
474 if position < self._layout.pageCollection().pageCount():
475 self._layout.pageCollection().insertPage(new_page, position)
476 else:

```
494     # Add at the end
495     position = self._layout.pageCollection().pageCount()
        self._layout.pageCollection().addPage(new_page)

        doc = QDomDocument()
        el = doc.createElement("CopyItems")
        ctx = QgsReadWriteContext()
        repeat_page_items = self._layout.pageCollection().itemsOnPage(
            self._repeat_page_num
        )
        for item in repeat_page_items:
            item.writeXml(el, doc, ctx)
            doc.appendChild(el)

        # Clear element identifier references
        nodes = doc.elementsByTagName("LayoutItem")
        for n in range(nodes.count()):
            node = nodes.at(n)
            if node.isElement():
                node.toElement().removeAttribute("uuid")

        page_ref_point = self._layout.pageCollection().pagePositionToLayoutPosition(
            position, QgsLayoutPoint(0, 0)
        )
        _ = self._layout.addItemFromXml(el, doc, ctx, page_ref_point, True)

    return True
```

export_to_pdf

 **Code:**

export_to_pdf()

Exports the layout to a PDF file in the output directory using the layout name as the file name.

Returns:

Type Description

bool True if the layout was successfully exported else False.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/generator.py`

1216

1217

1218

1219

1220

1221

1222

1223

1224

1225

1226

1227

1228

1229

1230

1231

1232

1233

1234

 **Code:**

```
def export_to_pdf(self) -> bool:
    """Exports the layout to a PDF file in the output
    directory using the layout name as the file name.

    :returns: True if the layout was successfully exported else False.
    :rtype: bool
    """

    if self._layout is None or self._project is None or not self.output_
        return False

    exporter = QgsLayoutExporter(self._layout)
    pdf_path = f"{self.output_dir}/{self._layout.name()}.pdf"
    result = exporter.exportToPdf(pdf_path, QgsLayoutExporter.PdfExports
    if result == QgsLayoutExporter.ExportResult.Success:
        return True
    else:
        tr_msg = tr("Could not export layout to PDF")
        self._error_messages.append(f"{tr_msg} {pdf_path}.")
        return False
```

run **Code:**`run()`

Initiates the report generation process and returns a result which contains information on whether the process succeeded or failed.

Returns:

Type	Description
<code>ReportResult</code>	The result of the report generation process.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/generator.py`

```
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109
```

 **Code:**

```
def run(self) -> ReportResult:  
    """Initiates the report generation process and returns  
    a result which contains information on whether the  
    process succeeded or failed.  
  
    :returns: The result of the report generation process.  
    :rtype: ReportResult  
    """  
  
    try:  
        return self._run()  
    except Exception as ex:  
        # Last resort to capture general exceptions.  
        exc_info = "".join(traceback.TracebackException.from_exception(ex).format())  
        self._error_messages.append(exc_info)  
        return self._get_failed_result()
```

`set_label_font` classmethod **Code:**

```
set_label_font(label, size, bold=False, italic=False)
```

Set font properties of the given layout label item.

Parameters:

Name	Type	Description	Default
label	<code>QgsLayoutItemLabel</code>	Label item whose font properties will be updated.required	
size	<code>float</code>	Point size of the font.	required
bold	<code>bool</code>	True if font is to be bold, else False (default).	<code>False</code>
italic	<code>bool</code>	True if font is to be in italics, else False (default).	<code>False</code>

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/generator.py

803
804
805  **Code:**
806
807
808
809
810 @classmethod
811 def set_label_font(
812 cls,
813 label: QgsLayoutItemLabel,
814 size: float,
815 bold: bool = False,
816 italic: bool = False,
817):
818 """Set font properties of the given layout label item.
819
820 :param label: Label item whose font properties will
821 be updated.
822 :type label: QgsLayoutItemLabel
823
824 :param size: Point size of the font.
825 :type size: int
826
827 :param bold: True if font is to be bold, else
828 False (default).
829 :type bold: bool
830
831 :param italic: True if font is to be in italics, else
832 False (default).
833 :type italic: bool
834
835 """
836
837 font = get_report_font(size, bold, italic)
838 version = Qgis.versionInt()
839
840
841 # Text format size unit
842 if version < 33000:
843 unit_type = QgsUnitTypes.RenderUnit.RenderPoints

```
else:  
    unit_type = Qgis.RenderUnit.Points  
  
    # Label font setting option  
    if version < 32400:  
        label.setFont(font)  
    else:  
        txt_format = QgsTextFormat()  
        txt_format.setFont(font)  
        txt_format.setSize(size)  
        txt_format.setSizeUnit(unit_type)  
        label.setTextFormat(txt_format)  
  
    label.refresh()
```

ReportGeneratorTask

Code:

```
ReportGeneratorTask(description, context)
```

Bases: [QgsTask](#)

Proxy class for initiating the report generation process.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/generator.py

```
71
72      Code:
73
74
75
76
77

def __init__(self, description: str, context: ReportContext):
    super().__init__(description)
    self._context = context
    self._result = None
    self._generator = ReportGenerator(self._context, self._context.f
    self.layout_manager = QgsProject.instance().layoutManager()
    self.layout_manager.layoutAdded.connect(self._on_layout_added)
```

context 

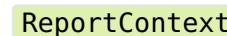
 **Code:**

context

Returns the report context used by the generator.

Returns:

Type **Description**

 Report context object used by the generator.

result 

 **Code:**

result

Returns the result object which contains information on whether the process succeeded or failed.

Returns:

Type	Description
------	-------------

ReportResult The result of the report generation process.

cancel

 **Code:**

cancel()

Cancel the report generation task.

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/generator.py](#)

98
99
100
101
102
103

```
def cancel(self):  
    """Cancel the report generation task."""  
    if self._context.feedback:  
        self._context.feedback.cancel()  
  
    super().cancel()
```

finished

 **Code:**

finished(result)

If successful, add the layout to the project.

Parameters:

Name	Type	Description	Default
result	bool	Flag indicating if the result of the report generation process. True if successful, else False.	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/generator.py

```
176  
177  
178  Code:  
179  
180  
181  
182  
183     def finished(self, result: bool):  
184         """If successful, add the layout to the project.  
185  
186             :param result: Flag indicating if the result of the  
187             report generation process. True if successful,  
188             else False.  
189             :type result: bool  
190             """  
191  
192         if len(self._result.messages) > 0:  
193             log(  
194                 f"Warnings and errors occurred when generating the "  
195                 f"report for {self._context.scenario.name} "  
196                 f"scenario. See details below:",  
197                 info=False,  
198             )  
199             for err in self._result.messages:  
200                 err_msg = f"{self._context.scenario.name} - {err}\n"  
201                 log(err_msg, info=False)  
202  
203         if result:  
204             log(  
205                 f"Successfully generated the report for "  
206                 f"{self._context.scenario.name} scenario."  
207             )  
208  
209         layout_path = self._generator.output_layout_path  
210         if not layout_path:  
211             log("Output layout could not be saved.", info=False)  
212         return  
213  
214     feedback = self._context.feedback
```

```
project = QgsProject.instance()
layout = _load_layout_from_file(layout_path, project)
if layout is None:
    log("Could not load layout from file.", info=False)
    return

# Zoom the extents of map items in the layout then export to PDF
self._zoom_map_items_to_current_extents(layout)
project.layoutManager().addLayout(layout)

if feedback is not None:
    feedback.setProgress(100)
```

run

 **Code:**

run()

Initiates the report generation process and returns a result indicating whether the process succeeded or failed.

Returns:

Type Description

bool True if the report generation process succeeded or False if failed.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/generator.py

105
106 Code:
107
108
109
110
111
112 def run(self) -> bool:
113 """Initiates the report generation process and returns
114 a result indicating whether the process succeeded or
115 failed.
116
117 :returns: True if the report generation process succeeded
118 or False if failed.
119 :rtype: bool
120 """
121
122 if self.isCanceled():
123 return False
124
125
126 if self._context.project_file:
127 self._result = self._generator.run()
128 else:
129 msg = tr("Unable to serialize current project for report g
130 msgs: typing.List[str] = [msg]
131 self._result = ReportResult(
132 False, self._context.scenario.uuid, "", tuple(msgs)
133)
134
135 return self._result.success

Layout Items

Custom CPLUS layout items.

CplusMapRepeatItem



```
CplusMapRepeatItem(*args, **kwargs)
```

Bases: [QgsLayoutItemShape](#)

Defines an outline area within a layout where map items containing NCS pathway or implementation model will be drawn.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/layout_items.py

27  **Code:**

28

29

30

31

32

33

34 def __init__(self, *args, **kwargs):

35 super().__init__(*args, **kwargs)

36 self.setShapeType(QgsLayoutItemShape.Shape.Rectangle)

37

38 # We shall use a frame so that it can be turned off / on

39 # using the item properties UI. The symbol is just a proxy.

40 # Symbol properties

41 symbol_props = {

42 "color": "229,182,54,0",

43 "style": "solid",

44 "outline_style": "dash",

45 "line_color": "132,192,68",

46 "outline_width": "0",

47 "joinstyle": "miter",

48 }

49 symbol = QgsFillSymbol.createSimple(symbol_props)

50 self.setSymbol(symbol)

51

52 self._model_component_type = kwargs.pop(

53 "model_component_type", ModelComponentType.UNKNOWN

54)

model_component_type `property` `writable`

 **Code:**

`model_component_type`

Gets the model component type associated with this map item i.e. NCS pathway or implementation model.

Returns:

Type Description

`Enum` Type of the model component.

`icon`

 **Code:**

`icon()`

Override for custom CPLUS map item.

- Source code in `/home/samwelij/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/layout_items.py`

83
84
85

```
def icon(self) -> QtGui.QIcon:  
    """Override for custom CPLUS map item."""  
    return FileUtils.get_icon("mLayoutItemMap_cplus.svg")
```

readPropertiesFromElement **Code:**

```
readPropertiesFromElement(element, document, context)
```

Override reading of item properties.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/layout_items.py`

95
96
97
98
99
100
101
102
103
104

```
def readPropertiesFromElement(self, element, document, context):
    """Override reading of item properties."""
    status = super().readPropertiesFromElement(element, document)
    if status:
        model_component_type = element.attribute("modelComponent")
        self._model_component_type = ModelComponentType.from_string(model_component_type)
    return status
```

type

 **Code:**

```
type()
```

Return item's unique type identifier.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/layout_items.py

71
72
73

 **Code:**

```
def type(self):
    """Return item's unique type identifier."""
    return CPLUS_MAP_REPEAT_ITEM_TYPE
```

visibleName

 **Code:**

```
visibleName()
```

Override for visible name of the item.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/layout_items.py

75
76
77

 **Code:**

```
def visibleName(self) -> str:
    """Override for visible name of the item."""
    return tr("CPLUS Map Repeat Area Item")
```

visiblePluralName **Code:**`visiblePluralName()`

Override for plural name of the items.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/layout_items.py`

79
80
81

 **Code:**

```
def visiblePluralName(self) -> str:  
    """Override for plural name of the items."""  
    return tr("CPLUS Map Repeat Area Items")
```

writePropertiesToElement **Code:**`writePropertiesToElement(el, document, context)`

Override saving of item properties.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/layout_items.py

```
87
88  Code:
89
90
91
92
93

def writePropertiesToElement(self, el, document, context):
    """Override saving of item properties."""
    status = super().writePropertiesToElement(el, document, context)
    if status:
        el.setAttribute("modelComponentType", self._model_component_type)

    return status
```

CplusMapRepeatItemLayoutItemMetadata

 **Code:**

```
CplusMapRepeatItemLayoutItemMetadata()
```

Bases: **QgsLayoutItemAbstractMetadata**

Metadata info of the cplus map repeat item.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/layout_items.py

```
110
111  Code:

def __init__(self):
    super().__init__(CPLUS_MAP_REPEAT_ITEM_TYPE, tr("CPLUS Map Repeat Item"))
```

createItem

 **Code:**

```
createItem(layout)
```

Factory method that return the cplus map item.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/layout_items.py

113
114
115

 **Code:**

```
def createItem(self, layout) -> CplusMapRepeatItem:  
    """Factory method that return the cplus map item."""  
    return CplusMapRepeatItem(layout)
```

Report Manager

Registers custom report variables for layout design and handles report generation.

ReportManager

 **Code:**

ReportManager(parent=None)

Bases: **QObject**

Registers custom report variables for layout design and handles report generation.

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/manager.py](#)

```
40
41  Code:
42
43
44
45
46
47 def __init__(self, parent=None):
48     super().__init__(parent)
49     self._variable_register = LayoutVariableRegister()
50     self.report_name = tr("Scenario Analysis Report")
51
52     # Task id (value) indexed by scenario id (key)
53     self._report_tasks = {}
54
55     # Report results (value) indexed by scenario id (key)
56     self._report_results = {}
57
58     self.task_manager = QgsApplication.instance().taskManager()
59     self.task_manager.statusChanged.connect(self.on_task_status_changed)
60
61     self.root_output_dir = ""
```

variable_register `property` **Code:**`variable_register`

Get the instance of the variable register used for the management of variables.

Returns:

Type	Description
<code>LayoutVariableRegister</code>	The register for managing variables in report layout scope.

`create_report_context` `classmethod` **Code:**`create_report_context(scenario_result, feedback)`

Creates the report context for use in the report generator task.

Parameters:

Name	Type	Description	Default
<code>scenario_result</code>	<code>ScenarioResult</code>	Result of the scenario analysis.	required
<code>feedback</code>	<code>QgsFeedback</code>	Feedback object for reporting back to the main application.	required

Returns:

Type	Description
<code>ReportContext</code>	A report context object containing the information for generating the report else None if it could not be created.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/manager.py

269
270  **Code:**

271
272
273
274
275
276 @classmethod
277
278 def create_report_context(
279 cls, scenario_result: ScenarioResult, feedback: QgsFeedback
280) -> typing.Union[ReportContext, None]:
281 """Creates the report context for use in the report
282 generator task.
283
284 :param scenario_result: Result of the scenario analysis.
285 :type scenario_result: ScenarioResult
286
287 :param feedback: Feedback object for reporting back to the main
288 application.
289 :type feedback: QgsFeedback
290
291 :returns: A report context object containing the information
292 for generating the report else None if it could not be created.
293 :rtype: ReportContext
294 """
295
296 output_dir = os.path.normpath(scenario_result.scenario_directory)
297 if not output_dir or not Path(output_dir).exists():
298 log(f"Unable to generate the report. {output_dir} not found.\n")
299 return None
300
301 scenario_report_dir = os.path.normpath(f"{output_dir}/reports")
302 FileUtils.create_new_dir(scenario_report_dir)
303
304 project_file_path = f"{scenario_report_dir}/{scenario_result.scenario}
305 if os.path.exists(project_file_path):
306 counter = 1
307 while True:
308 project_file_path = f"{scenario_report_dir}/{scenario_result.s
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325

```
326         if not os.path.exists(project_file_path):
327             break
328         counter += 1
329
330
331     # Write project to file for use in the task since QgsProject
332     # instances are not thread safe.
333     storage_type = QgsProject.instance().filePathStorage()
334     QgsProject.instance().setFilePathStorage(Qgis.FilePathType.Absolute)
335     result = QgsProject.instance().write(project_file_path)
336     QgsProject.instance().setFilePathStorage(storage_type)
337
338
339     if not result:
340         return None
341
342
343     # Set base name for the layout and PDF file suffixed with a number
344     # depending on the number of runs.
345     layout_manager = QgsProject.instance().layoutManager()
346     counter = 1
347     context_name = ""
348     while True:
349         layout_name = f"{scenario_result.scenario.name} {counter}s"
350         matching_layout = layout_manager.layoutByName(layout_name)
351         if matching_layout is None:
352             context_name = layout_name
353             break
354         counter += 1
355
356
357     template_path = FileUtils.report_template_path()
358
359
360     return ReportContext(
361         template_path,
362         scenario_result.scenario,
363         context_name,
364         scenario_report_dir,
365         project_file_path,
366         feedback,
```

```
    scenario_result.output_layer_name,  
)
```

create_scenario_dir

 **Code:**

```
create_scenario_dir(scenario)
```

Creates an output directory (within BASE_DIR) for saving the analysis outputs for the given scenario.

Parameters:

Name	Type	Description	Default
scenario	Scenario	Reference scenario object.required	

Returns:

Type Description

str The absolute path to the output directory. If BASE_DIR does not exist, it will not create the directory and will return an empty string. If the current user does not have write permissions to the base directory, it will return an empty string.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/manager.py

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

200

201

202

203

 **Code:**

```
def create_scenario_dir(self, scenario: Scenario) -> str:
    """Creates an output directory (within BASE_DIR) for saving the
    analysis outputs for the given scenario.

    :param scenario: Reference scenario object.
    :type scenario: Scenario

    :returns: The absolute path to the output directory. If
    BASE_DIR does not exist, it will not create the directory and
    will return an empty string. If the current user does not have
    write permissions to the base directory, it will return an
    empty string.
    :rtype: str
    """

    if not self.root_output_dir:
        return ""

    output_path = Path(self.root_output_dir)
    if not output_path.exists():
        try:
            output_path.mkdir()
        except FileNotFoundError:
            msg = (
                "Missing parent directory when creating "
                "outputs subdirectory in the base directory."
            )
            log(msg)
            return ""

    scenario_path_str = f"{self.root_output_dir}/{str(scenario.uuid)}"
    scenario_output_path = Path(scenario_path_str)
```

```
if not scenario_output_path.exists():
    try:
        scenario_output_path.mkdir()
    except FileNotFoundError:
        msg = (
            "Missing parent directory when creating "
            "scenario subdirectory in the outputs directory."
        )
        log(msg)
        return ""

return scenario_path_str
```

generate

Code:

```
generate(scenario_result, feedback=None)
```

Initiates the report generation process using information resulting from the scenario analysis.

Parameters:

Name	Type	Description	Default
scenario_result	ScenarioResult	Contains details from the scenario analysis.	required
feedback	QgsFeedback	Feedback for reporting back to the main application. If one is not specified then the manager will create one for the context.	None

Returns:

Type	Description
ReportSubmitStatus	True if the report generation process was successfully submitted else False if a running process is re-submitted. Object also contains feedback object for report updating and cancellation.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/manager.py

205
206
207  **Code:**
208
209
210
211
212 def generate(
213 self, scenario_result: ScenarioResult, feedback: QgsFeedback = None
214) -> ReportSubmitStatus:
215 """Initiates the report generation process using information
216 resulting from the scenario analysis.
217
218 :param scenario_result: Contains details from the scenario analysis.
219 :type scenario_result: ScenarioResult
220
221 :param feedback: Feedback for reporting back to the main application.
222 If one is not specified then the manager will create one for the context.
223 :type feedback: QgsFeedback
224
225
226 :returns: True if the report generation process was successfully
227 submitted else False if a running process is re-submitted. Object
228 also contains feedback object for report updating and cancellation.
229 :rtype: ReportSubmitStatus
230 """
231
232 scenario = scenario_result.scenario
233
234
235 if not scenario_result.output_layer_name:
236 log("Layer name for output scenario is empty. Cannot generate reports.")
237 return ReportSubmitStatus(False, None)
238
239
240 if feedback is None:
241 feedback = QgsFeedback(self)
242
243
244 ctx = self.create_report_context(scenario_result, feedback)
245 if ctx is None:
246 log("Could not create report context. Check directory settings.")
247 return ReportSubmitStatus(False, None)

```
scenario_id = str(ctx.scenario.uuid)
if scenario_id in self._report_tasks:
    return ReportSubmitStatus(False, ctx.feedback)

msg_tr = tr("Generating report for")
description = f"{msg_tr} {ctx.scenario.name}"
report_task = ReportGeneratorTask(description, ctx)
task_id = self.task_manager.addTask(report_task)

self._report_tasks[scenario_id] = task_id

return ReportSubmitStatus(True, ctx.feedback)
```

on_task_status_changed

Code:

```
on_task_status_changed(task_id, status)
```

Slot raised when the status of a task has changed.

This function will emit when the report generation task has started or when it has completed successfully or terminated due to an error.

Parameters:

Name	Type	Description	Default
task_id	int	ID of the task.	required
status	TaskStatus	New task status.	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/manager.py

98
99
100  **Code:**
101
102
103
104
105 def on_task_status_changed(self, task_id: int, status: QgsTask.TaskStatus)
106 """Slot raised when the status of a task has changed.
107
108
109 This function will emit when the report generation task has started
110 or when it has completed successfully or terminated due to an error.
111
112 :param task_id: ID of the task.
113 :type task_id: int
114
115 :param status: New task status.
116 :type status: QgsTask.TaskStatus
117 """
118
119 scenario_id = self.scenario_by_task_id(task_id)
120
121
122
123
124
125
126
127 # Not related to CPLUS report or task
128 if not scenario_id:
129 return
130
131
132
133
134 if status == QgsTask.TaskStatus.Running:
135 self.generate_started.emit(scenario_id)
136
137
138 elif status == QgsTask.TaskStatus.Complete:
139 # Get result
140 task = self.task_manager.task(task_id)
141 result = task.result
142 if result is not None:
143 self._report_results[scenario_id] = result
144
145
146 # Remove task
147 self.remove_report_task(scenario_id)

```
self.generate_completed.emit(scenario_id)
```

open_layout_designer classmethod

 **Code:**

```
open_layout_designer(result)
```

Opens the analysis report in the layout designer. The layout needs to exist in the currently loaded project.

Parameters:

Name	Type	Description	Default
result	ReportResult	Result object from the report generation process.required	

Returns:

Type	Description
bool	True if the layout was successfully loaded, else False if the result from the generation process was False or if the layout does not exist in the current project.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/manager.py

339
340
341  **Code:**
342
343
344
345
346
347
348 @classmethod
349
350 def open_layout_designer(cls, result: ReportResult) -> bool:
351 """Opens the analysis report in the layout designer. The
352 layout needs to exist in the currently loaded project.
353
354 :param result: Result object from the report generation
355 process.
356 :type result: ReportResult
357
358 :returns: True if the layout was successfully loaded, else
359 False if the result from the generation process was False
360 or if the layout does not exist in the current project.
361 :rtype: bool
362 """
363
364 if not result.success:
365 return False
366
367 layout = QgsProject.instance().layoutManager().layoutByName(r
368 if layout is None:
369 return False
370
371 iface.openLayoutDesigner(layout)
372
373 return True

register_variables**Code:**

```
register_variables(layout)
```

Registers custom variables and their corresponding initial values in the layout.

Parameters:

Name	Type	Description	Default
layout	QgsPrintLayout	Layout object where the custom variables will be registered.required	
• Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/manager.py			

Code:

```
67
68
69
70
71
72
73
74
75 def register_variables(self, layout: QgsPrintLayout):
    """Registers custom variables and their corresponding
    initial values in the layout.

    :param layout: Layout object where the custom
    variables will be registered.
    :type layout: QgsPrintLayout
    """
    self._variable_register.register_variables(layout)
```

remove_report_task

 **Code:**

```
remove_report_task(scenario_id)
```

Remove report task associated with the given scenario.

Parameters:

Name	Type	Description	Default
scenario_id	str	Identified of the scenario whose report generation process is to be removed.	required

Returns:

Type **Description**

bool	True if the task has been successfully removed else False if there is no associated task for the given scenario.
------	--

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/manager.py

 **Code:**

```
131
132
133
134
135
136
137
138 def remove_report_task(self, scenario_id: str) -> bool:
139     """Remove report task associated with the given scenario.
140
141     :param scenario_id: Identified of the scenario whose report
142     generation process is to be removed.
143     :type scenario_id: str
144
145
146     :returns: True if the task has been successfully removed
147     else False if there is no associated task for the given
148     scenario.
149     :rtype: bool
150
151     """
152
153     if scenario_id not in self._report_tasks:
154         return False
155
156
157     task_id = self._report_tasks[scenario_id]
158     task = self.task_manager.task(task_id)
159     if task is None:
160         return False
161
162     if (
163         task.status() != QgsTask.TaskStatus.Complete
164         or task.status() != QgsTask.TaskStatus.Terminated
165     ):
166         task.cancel()
167
168     _ = self._report_tasks.pop(scenario_id)
169
170     return True
```

report_result

 **Code:**

```
report_result(scenario_id)
```

Gets the report result for the scenario with the given ID.

Parameters:

Name	Type	Description	Default
scenario_id	str	Identifier of the scenario whose report is to be retrieved.	required

Returns:

Type	Description
ReportResult	Result of the report generation process. Caller needs to check if the process was successful or there was an error by checking the status of the success attribute. For scenarios that had not been submitted for report generation, a None object will be returned.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/manager.py

```
250
251  Code:
252
253
254
255
256
257     def report_result(self, scenario_id: str) -> typing.Union[ReportResult, None]:
258         """Gets the report result for the scenario with the given ID.
259
260         :param scenario_id: Identifier of the scenario whose report is to
261             be retrieved.
262         :type scenario_id: str
263
264         :returns: Result of the report generation process. Caller needs to
265             check if the process was successful or there was an error by checking
266             the status of the `success` attribute. For scenarios that had not
267             been submitted for report generation, a None object will be
268             returned.
269         :rtype: ReportResult
270         """
271
272         if scenario_id not in self._report_results:
273             return None
274
275
276         return self._report_results[scenario_id]
```

scenario_by_task_id

```
scenarios_by_task_id(task_id)
```

Gets the scenario identifier for the report generation task with the given ID.

Parameters:

Name	Type	Description	Default
task_id	int	ID of the task whose corresponding scenario is to be retrieved.required	

Returns:**Type Description**

str Scenario identifier whose report is being generated by a process with the given task id or an empty string if there was no match.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/manager.py`

77
78  **Code:**
79
80
81
82
83
84
85 def scenario_by_task_id(self, task_id: int) -> str:
86 """Gets the scenario identifier for the report generation t
87 ask with the given ID.
88
89
90 :param task_id: ID of the task whose corresponding scenario
91 is to be retrieved.
92 :type task_id: int
93
94 :returns: Scenario identifier whose report is being generated
95 by a process with the given task id or an empty string if
96 there was no match.
97 :rtype: str
98 """
99
100 scenario_ids = [
101 sid for sid, tid in self._report_tasks.items() if tid == ta
102]
103 if len(scenario_ids) == 0:
104 return ""
105
106 return scenario_ids[0]

`view_pdf classmethod` **Code:**`view_pdf(result)`

Opens the analysis in the host's default PDF viewer.

Parameters:

Name	Type	Description	Default
result	ReportResult	Result object from the report generation process.required	

Returns:**Type Description**

bool	True if the PDF was successfully loaded, else False if the result from the generation process was False.
------	--

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/manager.py

```
364
365
366
367
368
369
370
371 @classmethod
372 def view_pdf(cls, result: ReportResult) -> bool:
373     """Opens the analysis in the host's default PDF viewer.
374
375     :param result: Result object from the report generation
376     process.
377     :type result: ReportResult
378
379     :returns: True if the PDF was successfully loaded, else
380     False if the result from the generation process was False.
381     :rtype: bool
382
383     """
384
385     if not result.success:
386         return False
387
388     pdf_url = QtCore.QUrl.fromLocalFile(result.pdf_path)
389     if pdf_url.isEmpty():
390         return False
391
392     return QtGui.QDesktopServices.openUrl(pdf_url)
```

Variable Register

Manages custom variable data for report design and generation.

CplusVariableInfo dataclass

Contains information about a CPLUS variable within a layout scope.

update_final_value



update_final_value(context)

Computes the final value of the variable to be used in the layout.

Default implementation does nothing.

Parameters:

Name	Type	Description	Default
context	ReportContext	Report context object used to compute the final variable value.	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/variables.py

```
28  
29  
30   Code:  
31  
32  
33  
34  
35  
36  
37  
38
```

```
def update_final_value(self, context: ReportContext):  
    """Computes the final value of the variable to be used  
    in the layout.  
  
    Default implementation does nothing.  
  
    :param context: Report context object used to compute the  
    final variable value.  
    :type context: ReportContext  
    """  
    pass
```

LayoutVariableRegister

```
   Code:
```

```
LayoutVariableRegister()
```

Manages variables and their corresponding values for use in layout design and report generation.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/variables.py

131
132
133 **Code:**

```
def __init__(self):
    self._var_infos = {}
    self._init_vars()
```

var_name_init_values  **Code:****var_name_init_values**

Creates a collection of variable names and their corresponding initial values.

Returns:**Type Description****dict** Collection of variable names and corresponding initial values.**variable_names**  **Code:****variable_names**

Gets the names of the registered variables.

Returns:**Type Description****list** A collection of registered variable names.

is_analysis_report

Code:

```
is_analysis_report(layout)
```

Checks whether the layout has been produced from a report generation process.

Parameters:

Name	Type	Description	Default
layout	QgsPrintLayout	Layout to check whether its from a report generation process.required	

Returns:

Type Description

bool True if the layout is from a report generation process, else False.

- Source code in [/home/samwel/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/variables.py](#)

272

273

274

275

276

277

278

279

280

281

282

283

284

Code:

```
def is_analysis_report(self, layout: QgsPrintLayout) -> bool:  
    """Checks whether the layout has been produced from a report  
    generation process.  
  
    :param layout: Layout to check whether its from a report  
    generation process.  
    :type layout: QgsPrintLayout  
  
    :returns: True if the layout is from a report generation  
    process, else False.  
    :rtype: bool  
    """  
    return layout.customProperty(self.VAR_CPLUS_REPORT_PROPERTY, F
```

register_variables**Code:**

```
register_variables(layout)
```

Registers custom variables and their corresponding initial values in the layout.

Parameters:

Name	Type	Description	Default
layout	QgsPrintLayout	Layout object where the custom variables will be registered.required	

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/variables.py

286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309

 **Code:**

```
def register_variables(self, layout: QgsPrintLayout):
    """Registers custom variables and their corresponding
    initial values in the layout.

    :param layout: Layout object where the custom
    variables will be registered.
    :type layout: QgsPrintLayout
    """
    # If layout from analysis process, do not register
    # the variables.
    if self.is_analysis_report(layout):
        return

    # Remove any duplicate cplus variable names and values
    var_names, var_values = self.remove_variables(layout)

    # Get cplus variable names and corresponding initial values
    var_name_init_values = self.var_name_init_values
    for var_name, init_value in var_name_init_values.items():
        var_names.append(var_name)
        var_values.append(init_value)

    layout.setCustomProperty(self.VAR_NAMES_PROPERTY, var_names)
    layout.setCustomProperty(self.VAR_VALUES_PROPERTY, var_values)
```

remove_var_name_in_collection classmethod **Code:**

```
remove_var_name_in_collection(  
    cplus_var_name, layout_var_names, layout_var_values  
)
```

Remove cplus variable name matches and corresponding values in the layout variable name/value mapping.

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/variables.py](#)

```
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247
```

 **Code:**

```
@classmethod  
def remove_var_name_in_collection(  
    cls,  
    cplus_var_name: str,  
    layout_var_names: typing.List[str],  
    layout_var_values: typing.List[str],  
):  
    """Remove cplus variable name matches and corresponding  
    values in the layout variable name/value mapping.  
    """  
    while cplus_var_name in layout_var_names:  
        idx = layout_var_names.index(cplus_var_name)  
        _ = layout_var_names.pop(idx)  
        _ = layout_var_values.pop(idx)
```

```
remove_variables
```

 **Code:**

```
remove_variables(layout)
```

Removes duplicate variable names from the layout, this is done prior to registering new ones.

Parameters:

Name	Type	Description	Default
layout	QgsPrintLayout	Layout whose cplus variables are to be removed.required	

Returns:

Type	Description
tuple	Tuple only containing non-cplus variable names and corresponding values respectively.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/variables.py

```
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270
```

 **Code:**

```
def remove_variables(  
    self, layout: QgsPrintLayout  
) -> typing.Tuple[typing.List, typing.List]:  
    """Removes duplicate variable names from the layout,  
    this is done prior to registering new ones.  
  
    :param layout: Layout whose cplus variables are to be removed.  
    :type layout: QgsPrintLayout  
  
    :returns: Tuple only containing non-cplus variable names  
    and corresponding values respectively.  
    :rtype: tuple  
    """  
  
    cplus_var_names = self.variable_names  
    var_names = layout.customProperty(self.VAR_NAMES_PROPERTY, list())  
    var_values = layout.customProperty(self.VAR_VALUES_PROPERTY, list())  
  
    # Remove only cplus variable names and values  
    for cvn in cplus_var_names:  
        self.remove_var_name_in_collection(cvn, var_names, var_values)  
  
    return var_names, var_values
```

set_report_flag

 **Code:**

```
set_report_flag(layout)
```

Set a flag indicating that the layout has been produced from a report generation process.

Parameters:

Name	Type	Description	Default
layout	QgsPrintLayout	Layout to add the flag as a custom property.required	
• Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/variables.py			

```
311  
312  
313  
314  
315  
316  
317  
318
```

 **Code:**

```
def set_report_flag(self, layout: QgsPrintLayout):  
    """Set a flag indicating that the layout has been produced  
    from a report generation process.  
  
    :param layout: Layout to add the flag as a custom property.  
    :type layout: QgsPrintLayout  
    """  
    layout.setCustomProperty(self.VAR_CPLUS_REPORT_PROPERTY, True)
```

update_variables

 **Code:**

```
update_variables(layout, context)
```

Update the values for the CPLUS variables in the layout.

Parameters:

Name	Type	Description	Default
layout	QgsPrintLayout	Layout object whose CPLUS variable values will be updated. required	
context	ReportContext	Context object containing the report information that will be required used for computing the final value of the variable during the report generation process.	

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/variables.py`

320
321  **Code:**
322
323
324
325
326
327 def update_variables(self, layout: QgsPrintLayout, context: ReportContext):
328 """Update the values for the CPLUS variables in the layout.
329
330 :param layout: Layout object whose CPLUS variable values
331 will be updated.
332 :type layout: QgsPrintLayout
333
334 :param context: Context object containing the report information that
335 will be used for computing the final value of the variable during
336 the report generation process.
337 :type context: ReportContext
338 """
339
340 exp_scope = QgsExpressionContextUtils.layoutScope(layout)
341 var_names = exp_scope.variableNames()
342 var_values = []
343 vn = list(self._var_infos.keys())
344 for name in var_names:
345 if name in self._var_infos:
346 var_info = self._var_infos[name]
347 var_info.update_final_value(context)
348 var_values.append(var_info.final_value)
349 else:
350 if not exp_scope.hasVariable(name):
351 continue
352 value = exp_scope.variable(name)
353 var_values.append(value)
354
355 layout.setCustomProperty(self.VAR_NAMES_PROPERTY, var_names)
356 layout.setCustomProperty(self.VAR_VALUES_PROPERTY, var_values)
357 layout.refresh()

ScenarioDescriptionVariableInfo `dataclass`**Bases:** `CplusVariableInfo`

Metadata for a scenario description variable.

`update_final_value` **Code:**`update_final_value(context)`

Set the scenario description.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/variables.py`

116
117
118 **Code:**

```
def update_final_value(self, context: ReportContext):  
    """Set the scenario description."""  
    self.final_value = context.scenario.description
```

ScenarioNameVariableInfo `dataclass`**Bases:** `CplusVariableInfo`

Metadata for a scenario name variable.

`update_final_value` **Code:**`update_final_value(context)`

Set the scenario name.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/variables.py

94
95
96 **Code:**

```
def update_final_value(self, context: ReportContext):  
    """Set the scenario name."""  
    self.final_value = context.scenario.name
```

SettingsVariableInfo 

Bases: CplusVariableInfo

Metadata for a settings-related variable.

update_final_value

 **Code:**

```
update_final_value(context)
```

Computes the final value of the variable to be used in the layout.

Fetches the latest settings value.

Parameters:

Name	Type	Description	Default
context	ReportContext	Report context object used to compute the final variable value.	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/variables.py

```
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74
```

 **Code:**

```
def update_final_value(self, context: ReportContext):  
    """Computes the final value of the variable to be used  
    in the layout.
```

Fetches the latest settings value.

:param context: Report context object used to compute the final variable value.

:type context: ReportContext
"""

```
self.final_value = self._get_setting_value()
```

create_bulleted_text

 **Code:**

```
create_bulleted_text(main_text, bulleted_items)
```

Returns string containing text and bulleted/dashed text below it.

Parameters:

Name	Type	Description	Default
main_text	str	Primary non-bulleted text.	required
bulleted_items	List[str]	List containing bulleted items that will be rendered below required the main text.	required

Returns:**Type Description**

str Text containing primary text with bulleted items below it.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/reports/variables.py

```
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372
```

 **Code:**

```
def create_bulleted_text(main_text: str, bulleted_items: typing.List[  
    """Returns string containing text and bulleted/dashed text  
    below it.  
  
    :param main_text: Primary non-bulleted text.  
    :type main_text: str  
  
    :param bulleted_items: List containing bulleted items that will  
    be rendered below the main text.  
    :type bulleted_items: list  
  
    :returns: Text containing primary text with bulleted items  
    below it.  
    :rtype: str  
    """  
    bulleted_items = "\n- ".join(bulleted_items)  
  
    if not main_text:  
        return f"- {bulleted_items}"  
  
    return f"{main_text}\n- {bulleted_items}"
```

Pilot Extent Check

Checks if a given extent is within the pilot area of interest.

extent_within_pilot

Code:

```
extent_within_pilot(new_extent, source_crs=None)
```

Checks if the extent is within the pilot area.

Parameters:

Name	Type	Description	Default
new_extent	QgsRectangle	Extent to check if within the pilot area.	required
source_crs	QgsCoordinateReferenceSystem	Source coordinate reference system, if not specified then it will default to the project reference system. It reproject to WGS84 which is what is used for the pilot extent.	None

Returns:

Type Description

bool True if the current map canvas extent is within the pilot area, else False.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/lib/extent_check.py

 **Code:**

```
16
17
18
19
20
21
22
23 def extent_within_pilot(
24     new_extent: QgsRectangle, source_crs: QgsCoordinateReferenceSystem = None
25 ) -> bool:
26     """Checks if the extent is within the pilot area.
27
28     :param new_extent: Extent to check if within the pilot area.
29     :type new_extent: QgsRectangle
30
31     :param source_crs: Source coordinate reference system, if not specified then
32     it will default to the project reference system. It reproject to WGS84
33     which is what is used for the pilot extent.
34     :type source_crs: QgsCoordinateReferenceSystem
35
36     :returns: True if the current map canvas extent is within the
37     pilot area, else False.
38     :rtype: bool
39     """
40
41     if source_crs is None:
42         source_crs = QgsProject.instance().crs()
43
44     extent_list = PILOT_AREA_EXTENT["coordinates"]
45     pilot_extent = QgsRectangle(
46         extent_list[0], extent_list[2], extent_list[1], extent_list[3]
47     )
48
49     default_crs = QgsCoordinateReferenceSystem.fromEpsgId(DEFAULT_CRS_ID)
50     if default_crs != source_crs:
51         coordinate_xform = QgsCoordinateTransform(
52             source_crs, default_crs, QgsProject.instance()
53         )
54     new_extent = coordinate_xform.transformBoundingBox(new_extent)
```

```
return pilot_extent.contains(new_extent)
```

1.5.2 Models

Model base

QGIS CPLUS plugin models.

BaseModelComponent `dataclass`

Base class for common model item properties.

`__eq__`

 **Code:**

`__eq__(other)`

Test equality of object with another BaseModelComponent object using the attributes.

Parameters:

Name	Type	Description	Default
other	BaseModelComponent	BaseModelComponent object to compare with this object.	required

Returns:

Type	Description
bool	True if the all the attribute values match, else False.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py

 **Code:**

```
68
69
70
71
72
73
74
75 def __eq__(self, other: "BaseModelComponent") -> bool:
76     """Test equality of object with another BaseModelComponent
77     object using the attributes.
78
79     :param other: BaseModelComponent object to compare with this object.
80     :type other: BaseModelComponent
81
82     :returns: True if the all the attribute values match, else False.
83     :rtype: bool
84
85     """
86
87     if self.uuid != other.uuid:
88         return False
89
90     if self.name != other.name:
91         return False
92
93     if self.description != other.description:
94         return False
95
96     return True
```

ImplementationModel `[dataclass]`

Bases: `LayerModelComponent`

Contains information about the implementation model for a scenario. If the layer has been set then it will not be possible to add NCS pathways unless the layer is cleared. Priority will be given to the layer property.

__post_init__ **Code:**__post_init__()

Pre-checks on initialization.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py

```
329
330  Code:
331
332
333
334
335
336 def __post_init__(self):
337     """Pre-checks on initialization."""
338     super().__post_init__()
339
340     # Ensure there are no duplicate pathways.
341     uuids = [str(p.uuid) for p in self.pathways]
342
343
344     if len(set(uuids)) != len(uuids):
345         msg = "Duplicate pathways found in implementation model"
346         raise ValueError(f"{msg} {self.name}.")
347
348
349     # Reset pathways if layer has also been set.
350     if self.to_map_layer() is not None and len(self.pathways) > 0:
351         self.pathways = []
```

add_ncs_pathway

Code:

```
add_ncs_pathway(ncs)
```

Adds an NCS pathway object to the collection.

Parameters:

Name	Type	Description	Default
ncs	NcsPathway	NCS pathway to be added to the model.	required

Returns:

Type	Description
bool	True if the NCS pathway was successfully added, else False if there was an existing NCS pathway object with a similar UUID or the layer property had already been set.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py

```
359
360
361  Code:
362
363
364
365
366     def add_ncs_pathway(self, ncs: NcsPathway) -> bool:
367         """Adds an NCS pathway object to the collection.
368
369         :param ncs: NCS pathway to be added to the model.
370         :type ncs: NcsPathway
371
372         :returns: True if the NCS pathway was successfully added, else False
373         if there was an existing NCS pathway object with a similar UUID or
374         the layer property had already been set.
375
376         """
377
378
379         if not ncs.is_valid():
380             return False
381
382         if self.contains_pathway(str(ncs.uuid)):
383             return False
384
385         self.pathways.append(ncs)
386
387         return True
```

clear_layer

 **Code:**

```
clear_layer()
```

Removes a reference to the layer URI defined in the path attribute.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py`

380
381
382

 **Code:**

```
def clear_layer(self):
    """Removes a reference to the layer URI defined in the path attribute."""
    self.path = ""
```

contains_pathway

 **Code:**

```
contains_pathway(pathway_uuid)
```

Checks if there is an NCS pathway matching the given UUID.

Parameters:

Name	Type	Description	Default
<code>pathway_uuid</code>	<code>str</code>	UUID to search for in the collection.	required

Returns:

Type **Description**

`bool` True if there is a matching NCS pathway, else False.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py

344
345
346  **Code:**
347
348
349
350
351 def contains_pathway(self, pathway_uuid: str) -> bool:
352 """Checks if there is an NCS pathway matching the given UUID.
353
354 :param pathway_uuid: UUID to search for in the collection.
355 :type pathway_uuid: str
356
357 :returns: True if there is a matching NCS pathway, else False.
358 :rtype: bool
359 """
360
361 ncs_pathway = self.pathway_by_uuid(pathway_uuid)
362 if ncs_pathway is None:
363 return False
364
365 return True

is_pwls_valid

 **Code:**

```
is_pwls_valid()
```

Checks if the priority layers are valid.

Returns:

Type Description

bool True if all priority layers are valid, else False if even one is invalid. If there are no priority layers defined, it will always return True.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py

```
431
432
433  Code:
434
435
436
437
438     def is_pwls_valid(self) -> bool:
439         """Checks if the priority layers are valid.
440
441         :returns: True if all priority layers are valid, else False if
442         even one is invalid. If there are no priority layers defined, it will
443         always return True.
444         :rtype: bool
445         """
446
447         is_valid = True
448         for cl in self.pw_layers():
449             if not cl.isValid():
450                 is_valid = False
451                 break
452
453
454         return is_valid
```

is_valid

 **Code:**

is_valid()

Includes an additional check to assert if NCS pathways have been specified if the layer has not been set or is not valid.

Does not check for validity of individual NCS pathways in the collection.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py

```
447
448     Code:
449
450
451
452
453
454     def is_valid(self) -> bool:
455         """Includes an additional check to assert if NCS pathways have
456         been specified if the layer has not been set or is not valid.
457
458         Does not check for validity of individual NCS pathways in the
459         collection.
460
461         """
462
463         if self.to_map_layer() is not None:
464             return super().is_valid()
465         else:
466             if len(self.pathways) == 0:
467                 return False
468
469                 if not self.is_pwls_valid():
470                     return False
471
472             return True
```

model_color_ramp

Code:

```
model_color_ramp()
```

Create a color ramp for styling the implementation layer resulting from a scenario run.

Returns:

Type	Description
QgsColorRamp	A color ramp for styling the implementation layer or None if there was no definition found.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py`

512
513 Code:
514
515
516
517
518
519 def model_color_ramp(self) -> typing.Union[QgsColorRamp, None]:
520 """Create a color ramp for styling the implementation layer resulting
521 from a scenario run.
522
523 :returns: A color ramp for styling the implementation layer or None
524 if there was no definition found.
525 :rtype: QgsColorRamp
526 """
527 model_layer_info = self.model_layer_style_info()
528 if len(model_layer_info) == 0:
529 return None
530
531 ramp_info = model_layer_info.get(COLOR_RAMP_PROPERTIES_ATTRIBUTE, None)
532 if ramp_info is None or len(ramp_info) == 0:
533 return None
534
535 ramp_type = model_layer_info.get(COLOR_RAMP_TYPE_ATTRIBUTE, None)
536 if ramp_type is None:
537 return None
538
539 # New ramp types will need to be added here manually
540 if ramp_type == QgsColorBrewerColorRamp.typeString():
541 return QgsColorBrewerColorRamp.create(ramp_info)
542 elif ramp_type == QgsCptCityColorRamp.typeString():
543 return QgsCptCityColorRamp.create(ramp_info)
544 elif ramp_type == QgsGradientColorRamp.typeString():
545 return QgsGradientColorRamp.create(ramp_info)
546 elif ramp_type == QgsLimitedRandomColorRamp.typeString():
547 return QgsLimitedRandomColorRamp.create(ramp_info)
548 elif ramp_type == QgsPresetSchemeColorRamp.typeString():
549 return QgsPresetSchemeColorRamp.create(ramp_info)

```
elif ramp_type == QgsRandomColorRamp.typeString():
    return QgsRandomColorRamp()

return None
```

model_layer_style_info

Code:

```
model_layer_style_info()
```

Returns the color ramp properties for styling the implementation layer resulting from a scenario run.

Returns:

Type Description

dict Color ramp properties for the implementation model styling or an empty dictionary if there was no definition found in the root style.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py

```
482
483      Code:
484
485
486
487
488
489     def model_layer_style_info(self) -> dict:
490         """Returns the color ramp properties for styling the implementation
491         layer resulting from a scenario run.
492
493         :returns: Color ramp properties for the implementation model styling
494         or an empty dictionary if there was no definition found in the root
495         style.
496         :rtype: dict
497         """
498
499         if (
500             len(self.layer_styles) == 0
501             or IM_LAYER_STYLE_ATTRIBUTE not in self.layer_styles
502         ):
503             return dict()
504
505
506         return self.layer_styles[IM_LAYER_STYLE_ATTRIBUTE]
```

pathway_by_uuid

 **Code:**

```
pathway_by_uuid(pathway_uuid)
```

Returns an NCS pathway matching the given UUID.

Parameters:

Name	Type	Description	Default
pathway_uuid	str	UUID for the NCS pathway to retrieve.	required

Returns:

Type	Description
NcsPathway	NCS pathway object matching the given UUID else None if not found.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py

404
405 **Code:**
406
407
408
409
410
411 def pathway_by_uuid(self, pathway_uuid: str) -> typing.Union[NcsPathway
412 """Returns an NCS pathway matching the given UUID.
413
414
415 :param pathway_uuid: UUID for the NCS pathway to retrieve.
416 :type pathway_uuid: str
417
418 :returns: NCS pathway object matching the given UUID else None if
419 not found.
420 :rtype: NcsPathway
421
422 """
423
424 pathways = [p for p in self.pathways if str(p.uuid) == pathway_uuid]
425
426
427 if len(pathways) == 0:
428 return None
429
430
431 return pathways[0]

pw_layers

Code:

pw_layers()

Returns the list of priority weighting layers wdefined under the :py:attr: `~priority_layers` attribute.

Returns:

Type	Description
<code>list</code>	Priority layers for the implementation or an empty list if the path is not defined.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py`

```
421
422
423
424
425
426
427
428
429
```

 **Code:**

```
def pw_layers(self) -> typing.List[QgsRasterLayer]:
    """Returns the list of priority weighting layers wdefined under
    the :py:attr:`~priority_layers` attribute.

    :returns: Priority layers for the implementation or an empty list
    if the path is not defined.
    :rtype: list
    """

    return [QgsRasterLayer(layer.get("path")) for layer in self.priorit
```

remove_ncs_pathway

 **Code:**

```
remove_ncs_pathway(pathway_uuid)
```

Removes the NCS pathway with a matching UUID from the collection.

Parameters:

Name	Type	Description	Default
<code>pathway_uuid</code>	<code>str</code>	UUID for the NCS pathway to be removed.required	

Returns:**Type Description**

- bool** True if the NCS pathway object was successfully removed, else False if there is no object matching the given UUID.
- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py](#)

384
385 💻 **Code:**
386
387
388
389
390
391 **def remove_ncs_pathway(self, pathway_uuid: str) -> bool:**
392 """Removes the NCS pathway with a matching UUID from the collection.
393
394
395 :param pathway_uuid: UUID for the NCS pathway to be removed.
396 :type pathway_uuid: str
397
398 :returns: True if the NCS pathway object was successfully removed,
399 else False if there is no object matching the given UUID.
400 :rtype: bool
401
402 """
403
404 idxs = [i for i, p in enumerate(self.pathways) if str(p.uuid) == pathway_uuid]
405
406
407 if len(idxs) == 0:
408 return False
409
410 rem_idx = idxs[0]
411 _ = self.pathways.pop(rem_idx)
412
413 return True

scenario_fill_symbol

Code:

```
scenario_fill_symbol()
```

Creates a fill symbol for the implementation model in the scenario.

Returns:

Type	Description
------	-------------

QgsFillSymbol	Fill symbol for the implementation model in the scenario or None if there was no definition found.
----------------------	--

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py](#)

```
499
500
501  Code:
502
503
504
505
506
507     def scenario_fill_symbol(self) -> typing.Union[QgsFillSymbol, None]:
508         """Creates a fill symbol for the implementation model in the scenario.
509
510             :returns: Fill symbol for the implementation model in the scenario
511             or None if there was no definition found.
512             :rtype: QgsFillSymbol
513
514
515             scenario_style_info = self.scenario_layer_style_info()
516             if len(scenario_style_info) == 0:
517                 return None
518
519
520             return QgsFillSymbol.createSimple(scenario_style_info)
```

scenario_layer_style_info

Code:

```
scenario_layer_style_info()
```

Returns the fill symbol properties for styling the implementation layer in the final scenario result.

Returns:

Type Description

`dict` Fill symbol properties for the implementation model styling in the scenario layer or an empty dictionary if there was no definition found in the root style.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py

```
465  
466  
467  Code:  
468  
469  
470  
471  
472     def scenario_layer_style_info(self) -> dict:  
473         """Returns the fill symbol properties for styling the implementation  
474         layer in the final scenario result.  
475  
476         :returns: Fill symbol properties for the implementation model  
477         styling in the scenario layer or an empty dictionary if there was  
478         no definition found in the root style.  
479         :rtype: dict  
480         """  
481  
482         if (br/>483             len(self.layer_styles) == 0  
484             or IM_SCENARIO_STYLE_ATTRIBUTE not in self.layer_styles  
485         ):  
486             return dict()  
487  
488         return self.layer_styles[IM_SCENARIO_STYLE_ATTRIBUTE]
```

LayerModelComponent

Bases:  **BaseModelComponent**

Base class for model components that support a map layer.

__eq__

 **Code:**

```
__eq__(other)
```

Uses BaseModelComponent equality test rather than what the dataclass default implementation will provide.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py`

```
198  
199  
200  
201  
202
```

 **Code:**

```
def __eq__(self, other) -> bool:  
    """Uses BaseModelComponent equality test rather than  
    what the dataclass default implementation will provide.  
    """  
    return super().__eq__(other)
```

`__post_init__`

 **Code:**

`__post_init__()`

Try to set the layer and layer type properties.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py`

```
141  
142  
143
```

 **Code:**

```
def __post_init__(self):  
    """Try to set the layer and layer type properties."""  
    self.update_layer_type()
```

is_valid **Code:**`is_valid()`

Checks if the corresponding map layer is valid.

Returns:**Type Description**

`bool` True if the map layer is valid, else False if map layer is invalid or of None type.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py`

```
185
186
187
188
189
190
191
192     def is_valid(self) -> bool:
193         """Checks if the corresponding map layer is valid.
194
195         :returns: True if the map layer is valid, else False if map layer is
196         invalid or of None type.
197         :rtype: bool
198
199         """
200
201         layer = self.to_map_layer()
202         if layer is None:
203             return False
204
205
206         return layer.isValid()
```



Constructs a map layer from the specified path.

It will first check if the layer property has been set else try to construct the layer from the path else return None.

Returns:

Type	Description
------	-------------

QgsMapLayer	Map layer corresponding to the set layer property or specified path.
-------------	--

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py

162
163  **Code:**

164
165
166
167
168
169 def to_map_layer(self) -> typing.Union[QgsMapLayer, None]:
170 """Constructs a map layer from the specified path.
171
172 It will first check if the layer property has been set
173 else try to construct the layer from the path else return
174 None.
175
176 :returns: Map layer corresponding to the set layer
177 property or specified path.
178 :rtype: QgsMapLayer
179 """
180
181 if not os.path.exists(self.path):
182 return None
183
184
185 layer = None
186 if self.layer_type == LayerType.RASTER:
187 layer = QgsRasterLayer(self.path, self.name)
188
189 elif self.layer_type == LayerType.VECTOR:
190 layer = QgsVectorLayer(self.path, self.name)
191
192 return layer

update_layer_type

Code:

```
update_layer_type()
```

Update the layer type if either the layer or path properties have been set.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py

```
145
146     def update_layer_type(self):
147         """
148             Update the layer type if either the layer or
149             path properties have been set.
150
151
152             layer = self.to_map_layer()
153             if layer is None:
154                 return
155
156
157             if not layer.isValid():
158                 return
159
160
161             if isinstance(layer, QgsRasterLayer):
162                 self.layer_type = LayerType.RASTER
163
164             elif isinstance(layer, QgsVectorLayer):
165                 self.layer_type = LayerType.VECTOR
```

LayerType

Bases: IntEnum

QGIS spatial layer type.

ModelComponentType

Bases: `Enum`

Type of model component i.e. NCS pathway or implementation model.

`from_string` | `staticmethod`

 **Code:**

```
from_string(str_enum)
```

Creates an enum from the corresponding string equivalent.

Parameters:

Name	Type	Description	Default
<code>str_enum</code>	<code>str</code>	String representing the model component type.required	

Returns:

Type	Description
<code>ModelComponentType</code>	Component type enum corresponding to the given string else unknown if not found.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py

```
112
113      Code:
114
115
116
117
118
119     @staticmethod
120
121     def from_string(str_enum: str) -> "ModelComponentType":
122         """Creates an enum from the corresponding string equivalent.
123
124         :param str_enum: String representing the model component type.
125         :type str_enum: str
126
127
128
129         :returns: Component type enum corresponding to the given
130         string else unknown if not found.
131         :rtype: ModelComponentType
132
133
134         if str_enum.lower() == "ncs_pathway":
135             return ModelComponentType.NCS_PATHWAY
136         elif str_enum.lower() == "implementation_model":
137             return ModelComponentType.IMPLEMENTATION_MODEL
138
139
140         return ModelComponentType.UNKNOWN
```

NcsPathway 

Bases: LayerModelComponent

Contains information about an NCS pathway layer.

__eq__

 **Code:**

__eq__(other)

Test equality of NcsPathway object with another NcsPathway object using the attributes.

Excludes testing the map layer for equality.

Parameters:

Name	Type	Description	Default
other	NcsPathway	NcsPathway object to compare with this object.required	

Returns:

Type	Description
bool	True if all the attribute values match, else False.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py

225
226  **Code:**

227
228
229
230
231
232 def __eq__(self, other: "NcsPathway") -> bool:
233 """Test equality of NcsPathway object with another
234 NcsPathway object using the attributes.
235
236 Excludes testing the map layer for equality.
237
238 :param other: NcsPathway object to compare with this object.
239 :type other: NcsPathway
240
241 :returns: True if all the attribute values match, else False.
242 :rtype: bool
243 """
244 base_equality = super().__eq__(other)
245 if not base_equality:
246 return False
247
248 if self.path != other.path:
249 return False
250
251 if self.layer_type != other.layer_type:
252 return False
253
254 if self.user_defined != other.user_defined:
255 return False
256
257 return True

add_carbon_path

Code:

```
add_carbon_path(carbon_path)
```

Add a carbon layer path.

Checks if the path has already been defined or if it exists in the file system.

Returns:

Type Description

`bool` True if the carbon layer path was successfully added, else False if the path has already been defined or does not exist in the file system.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py

```
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271
```

 **Code:**

```
def add_carbon_path(self, carbon_path: str) -> bool:  
    """Add a carbon layer path.  
  
    Checks if the path has already been defined or if it exists  
    in the file system.  
  
    :returns: True if the carbon layer path was successfully  
    added, else False if the path has already been defined  
    or does not exist in the file system.  
    :rtype: bool  
    """  
  
    if carbon_path in self.carbon_paths:  
        return False  
  
    if not os.path.exists(carbon_path):  
        return False  
  
    self.carbon_paths.append(carbon_path)  
  
    return True
```

carbon_layers **Code:**

```
carbon_layers()
```

Returns the list of carbon layers whose path is defined under the :py:attr: `~carbon_paths` attribute.

The caller should check the validity of the layers or use :py:meth: `~is_carbon_valid` function.

Returns:**Type Description**

list Carbon layers for the NCS pathway or an empty list if the path is not defined.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py`

273
274
275
276
277
278
279
280
281
282
283
284

 **Code:**

```
def carbon_layers(self) -> typing.List[QgsRasterLayer]:  
    """Returns the list of carbon layers whose path is defined under  
    the :py:attr:`~carbon_paths` attribute.
```

The caller should check the validity of the layers or use :py:meth: `~is_carbon_valid` function.

:returns: Carbon layers for the NCS pathway or an empty list if the path is not defined.

:rtype: list

"""

```
return [QgsRasterLayer(carbon_path) for carbon_path in self.carbon
```

is_carbon_valid **Code:**

```
is_carbon_valid()
```

Checks if the carbon layers are valid.

Returns:**Type Description**

`bool` True if all carbon layers are valid, else False if even one is invalid. If there are no carbon layers defined, it will always return True.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py`

```
286
287  Code:
288
289
290
291
292
293     def is_carbon_valid(self) -> bool:
294         """Checks if the carbon layers are valid.
295
296         :returns: True if all carbon layers are valid, else False if
297         even one is invalid. If there are no carbon layers defined, it will
298         always return True.
299         :rtype: bool
300
301         """
302
303         is_valid = True
304         for cl in self.carbon_layers():
305             if not cl.isValid():
306                 is_valid = False
307                 break
308
309
310         return is_valid
```

`is_valid`

 **Code:**

```
is_valid()
```

Additional check to include validity of carbon layers.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/base.py

```
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312
```

 **Code:**

```
def is_valid(self) -> bool:  
    """Additional check to include validity of carbon layers."""  
    valid = super().is_valid()  
    if not valid:  
        return False  
  
    carbon_valid = self.is_carbon_valid()  
    if not carbon_valid:  
        return False  
  
    return True
```

PRIORITY_GROUP

Bases: **Enum**

Represents priority groups types

PriorityLayer **dataclass**

Bases: **BaseModelComponent**

Base class for model components storing priority weighting layers.

Scenario **dataclass**

Bases: **BaseModelComponent**

Object for the handling workflow scenario information.

ScenarioResult **dataclass**

Scenario result details.

ScenarioState

Bases: **Enum**

Defines scenario analysis process states

SpatialExtent **dataclass**

Extent object that stores the coordinates of the area of interest

Helpers

Helper functions for supporting model management.

clone_implementation_model

Code:

```
clone_implementation_model(implementation_model)
```

Creates a deep copy of the given implementation model.

Parameters:

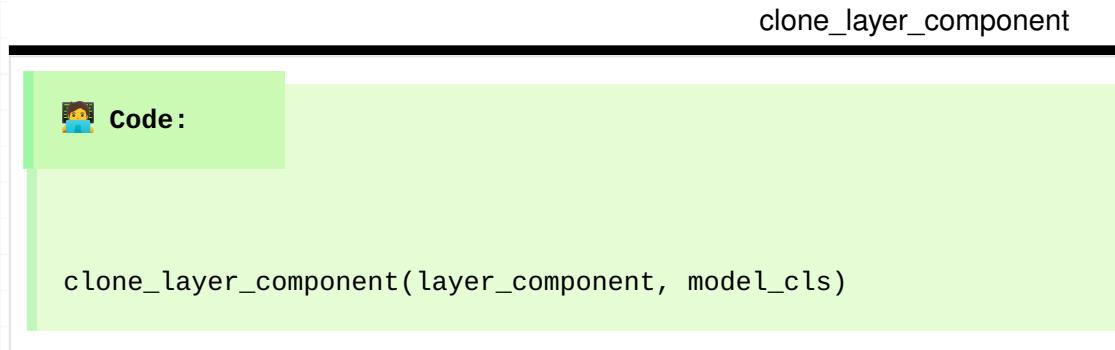
Name	Type	Description	Default
implementation_model	ImplementationModel	Implementation model to clone.required	

Returns:

Type	Description
ImplementationModel	A deep copy of the original implementation model object.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/helpers.py

```
294
295      Code:
296
297
298
299
300
301     def clone_implementation_model(
302         implementation_model: ImplementationModel,
303     ) -> ImplementationModel:
304         """Creates a deep copy of the given implementation model.
305
306         :param implementation_model: Implementation model to clone.
307         :type implementation_model: ImplementationModel
308
309         :returns: A deep copy of the original implementation model object.
310         :rtype: ImplementationModel
311
312         """
313
314         imp_model = clone_layer_component(implementation_model, ImplementationModel)
315         if imp_model is None:
316             return None
317
318
319         pathways = implementation_model.pathways
320         cloned_pathways = []
321         for p in pathways:
322             cloned_ncs = clone_ncs_pathway(p)
323             if cloned_ncs is not None:
324                 cloned_pathways.append(cloned_ncs)
325
326         imp_model.pathways = cloned_pathways
327
328         return imp_model
```



Clones a layer-based model component.

Parameters:

Name	Type	DescriptionDefault
<code>layer_component</code>	<code>LayerModelComponent</code>	Layer-based required model component to clone.
<code>model_cls</code>	<code>Callable[[UUID, str, str], LayerModelComponentType]</code>	Callable required class that will be created based on the input argument values from the dictionary.

Returns:

Type	Description
<code>LayerModelComponent</code>	A new instance of the cloned model component. It will return None if the input is not a layer-based model component.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/helpers.py

250
251 Code:
252
253
254
255
256
257 def clone_layer_component(
258 layer_component: LayerModelComponent,
259 model_cls: typing.Callable[[uuid.UUID, str, str], LayerModelComponent]
260) -> typing.Union[LayerModelComponent, None]:
261 """Clones a layer-based model component.
262
263 :param layer_component: Layer-based model component to clone.
264 :type layer_component: LayerModelComponent
265
266 :param model_cls: Callable class that will be created based on the
267 input argument values from the dictionary.
268 :type model_cls: LayerModelComponent
269
270 :returns: A new instance of the cloned model component. It
271 will return None if the input is not a layer-based model
272 component.
273 :rtype: LayerModelComponent
274 """
275
276 if not isinstance(layer_component, LayerModelComponent):
277 return None
278
279 cloned_component = model_cls(
280 layer_component.uuid, layer_component.name, layer_component.desc
281)
282
283 for f in fields(layer_component):
284 attr_val = getattr(layer_component, f.name)
285 setattr(cloned_component, f.name, attr_val)
286
287 return cloned_component

clone_ncs_pathway

Code:

```
clone_ncs_pathway(ncs)
```

Creates a deep copy of the given NCS pathway.

Parameters:

Name	Type	Description	Default
ncs	NcsPathway	NCS pathway to clone.required	

Returns:

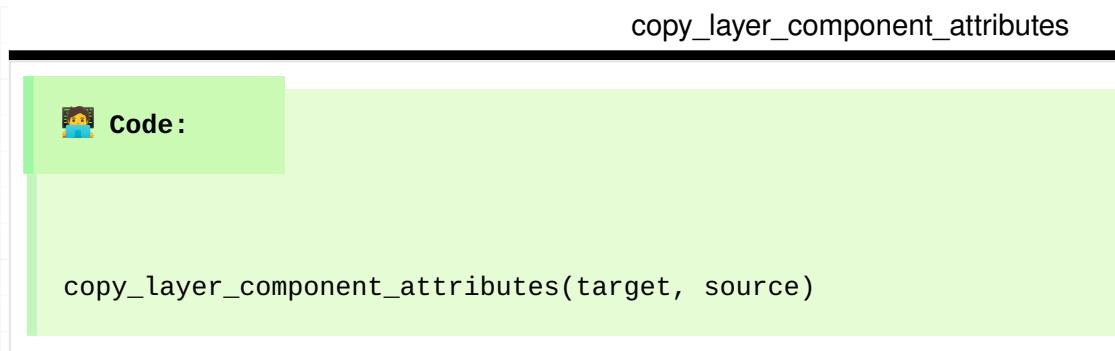
Type	Description
NcsPathway	A deep copy of the original NCS pathway object.

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/helpers.py](#)

```
282  
283  
284  
285  
286  
287  
288  
289  
290  
291
```

Code:

```
def clone_ncs_pathway(ncs: NcsPathway) -> NcsPathway:  
    """Creates a deep copy of the given NCS pathway.  
  
    :param ncs: NCS pathway to clone.  
    :type ncs: NcsPathway  
  
    :returns: A deep copy of the original NCS pathway object.  
    :rtype: NcsPathway  
    """  
    return clone_layer_component(ncs, NcsPathway)
```



Copies the attribute values of source to target. The uid attribute value is not copied as well as the layer attribute. However, for the latter, the path is copied.

Parameters:

Name	Type	Description	Default
target	LayerModelComponent	Target object whose attribute values will be updated. required	
source	LayerModelComponent	Source object whose attribute values will be copied to required the target.	

Returns:

Type	Description
LayerModelComponent	Target object containing the updated attribute values apart for the uid whose value will not change.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/helpers.py

321
322  **Code:**

323
324
325
326
327
328 def copy_layer_component_attributes(
329 target: LayerModelComponent, source: LayerModelComponent
330) -> LayerModelComponent:
331 """Copies the attribute values of source to target. The uuid
332 attribute value is not copied as well as the layer attribute.
333 However, for the latter, the path is copied.
334
335 :param target: Target object whose attribute values will be updated.
336 :type target: LayerModelComponent
337
338 :param source: Source object whose attribute values will be copied to
339 the target.
340 :type source: LayerModelComponent
341
342 :returns: Target object containing the updated attribute values apart
343 for the uuid whose value will not change.
344 :rtype: LayerModelComponent
345 """
346
347 if not isinstance(target, LayerModelComponent) or not isinstance(
348 source, LayerModelComponent
349):
350 raise TypeError(
351 "Source or target objects are not of type 'LayerModelComponent'"
352)
353
354
355
356 for f in fields(source):
357 # Exclude uuid
358 if f.name == UUID_ATTRIBUTE:
359 continue
360 attr_val = getattr(source, f.name)
361 setattr(target, f.name, attr_val)

```
# Force layer to be set/updated
target.update_layer_type()

return target
```

create_implementation_model

Code:

```
create_implementation_model(source_dict)
```

Factory method for creating an implementation model using attribute values defined in a dictionary.

Parameters:

Name	Type	Description	Default
source_dict	dict	Dictionary containing property values.required	

Returns:

Type	Description
ImplementationModel	Implementation model with property values set from the dictionary.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/helpers.py

168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191

 **Code:**

```
def create_implementation_model(source_dict) -> typing.Union[ImplementationModel, None]:
    """Factory method for creating an implementation model using attribute values defined in a dictionary.

    :param source_dict: Dictionary containing property values.
    :type source_dict: dict

    :returns: Implementation model with property values set from the dictionary.
    :rtype: ImplementationModel
    """

    implementation_model = create_layer_component(source_dict, ImplementationModel)
    if PRIORITY_LAYERS_SEGMENT in source_dict.keys():
        implementation_model.priority_layers = source_dict[PRIORITY_LAYERS_SEGMENT]

    # Set style
    if STYLE_ATTRIBUTE in source_dict.keys():
        implementation_model.layer_styles = source_dict[STYLE_ATTRIBUTE]

    # Set styling pixel value
    if PIXEL_VALUE_ATTRIBUTE in source_dict.keys():
        implementation_model.style_pixel_value = source_dict[PIXEL_VALUE_ATTRIBUTE]

    return implementation_model
```

create_layer_component

Code:

```
create_layer_component(source_dict, model_cls)
```

Factory method for creating a layer model component using attribute values defined in a dictionary.

Parameters:

Name	Type
source_dict	dict

model_cls Callable[[UUID, str, str, str, LayerType, bool], LayerModelComponentType]

Returns:

Type	Description
LayerModelComponent	Layer model component object with property values set from the dictionary.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/helpers.py

101
102  **Code:**

103
104
105
106
107
108 def create_layer_component(
109 source_dict,
110 model_cls: typing.Callable[
111 [uuid.UUID, str, str, str, LayerType, bool], LayerModelComponent
112],
113) -> typing.Union[LayerModelComponent, None]:
114 """Factory method for creating a layer model component using
115 attribute values defined in a dictionary.
116
117 :param source_dict: Dictionary containing property values.
118 :type source_dict: dict
119
120 :param model_cls: Callable class that will be created based on the
121 input argument values from the dictionary.
122 :type model_cls: LayerModelComponent
123
124 :returns: Layer model component object with property values set
125 from the dictionary.
126 :rtype: LayerModelComponent
127 """
128
129 if UUID_ATTRIBUTE not in source_dict:
130 return None
131
132
133 source_uuid = source_dict[UUID_ATTRIBUTE]
134 if isinstance(source_uuid, str):
135 source_uuid = uuid.UUID(source_uuid)
136
137 kwargs = {}
138 if PATH_ATTRIBUTE in source_dict:
139 kwargs[PATH_ATTRIBUTE] = source_dict[PATH_ATTRIBUTE]

```
if LAYER_TYPE_ATTRIBUTE in source_dict:  
    kwargs[LAYER_TYPE_ATTRIBUTE] = LayerType(int(source_dict[LAYER_T  
  
if USER_DEFINED_ATTRIBUTE in source_dict:  
    kwargs[USER_DEFINED_ATTRIBUTE] = bool(source_dict[USER_DEFINED_A  
  
return model_cls(  
    source_uuid,  
    source_dict[NAME_ATTRIBUTE],  
    source_dict[DESCRIPTION_ATTRIBUTE],  
    **kwargs  
)
```

create_model_component

Code:

```
create_model_component(source_dict, model_cls)
```

Factory method for creating and setting attribute values for a base model component object.

Parameters:

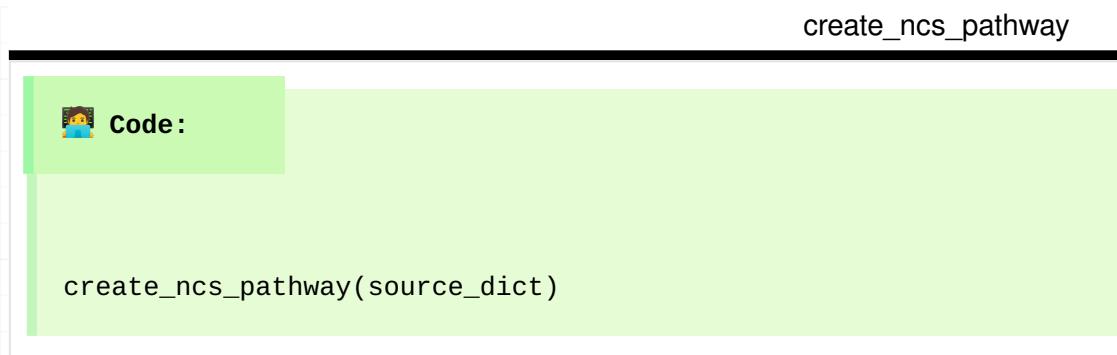
Name	Type	DescriptionDefault
source_dict	dict	Dictionary required containing attribute values.
model_cls	Callable[[UUID, str, str], BaseModelComponentType]	Callable required class that will be created based on the input argument values from the dictionary.

Returns:

Type	Description
BaseModelComponent	Base model component object with property values derived from the dictionary.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/helpers.py

73
74  **Code:**
75
76
77
78
79
80 def create_model_component(
81 source_dict: dict,
82 model_cls: typing.Callable[[uuid.UUID, str, str], BaseModelComponent]
83) -> typing.Union[BaseModelComponentType, None]:
84 """Factory method for creating and setting attribute values
85 for a base model component object.
86
87 :param source_dict: Dictionary containing attribute values.
88 :type source_dict: dict
89
90
91 :param model_cls: Callable class that will be created based on the
92 input argument values from the dictionary.
93 :type model_cls: BaseModelComponent
94
95
96 :returns: Base model component object with property values
97 derived from the dictionary.
98 :rtype: BaseModelComponent
99 """
100 if not issubclass(model_cls, BaseModelComponent):
101 return None
102
103 return model_cls(
104 uuid.UUID(source_dict[UUID_ATTRIBUTE]),
105 source_dict[NAME_ATTRIBUTE],
106 source_dict[DESCRIPTION_ATTRIBUTE],
107)



Factory method for creating an NcsPathway object using attribute values defined in a dictionary.

Parameters:

Name	Type	Description	Default
<code>source_dict</code>	<code>dict</code>	Dictionary containing property values.required	

Returns:

Type	Description
<code>NcsPathway</code>	NCS pathway object with property values set from the dictionary.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/helpers.py

```
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165
```

 **Code:**

```
def create_ncs_pathway(source_dict) -> typing.Union[NcsPathway, None]:  
    """Factory method for creating an NcsPathway object using  
    attribute values defined in a dictionary.  
  
    :param source_dict: Dictionary containing property values.  
    :type source_dict: dict  
  
    :returns: NCS pathway object with property values set  
    from the dictionary.  
    :rtype: NcsPathway  
    """  
  
    ncs = create_layer_component(source_dict, NcsPathway)  
  
    # We are checking because of the various iterations of the attributes  
    # in the NcsPathway class where some of these attributes might  
    # be missing.  
    if CARBON_PATHS_ATTRIBUTE in source_dict:  
        ncs.carbon_paths = source_dict[CARBON_PATHS_ATTRIBUTE]  
  
    return ncs
```

extent_to_project_crs_extent

 **Code:**

```
extent_to_project_crs_extent(spatial_extent, project=None)
```

Transforms SpatialExtent model to an QGIS extent based on the CRS of the given project.

Parameters:

Name	Type	Description	Default
spatial_extent	SpatialExtent	Spatial extent data model that defines the scenario required bounds.	
project	QgsProject	Project whose CRS will be used to determine the values of the output extent.	None

Returns:

Type	Description
QgsRectangle	Output extent in the project's CRS. If the input extent is invalid, this function will return None.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/helpers.py

385
386
387  **Code:**
388
389
390
391
392 def extent_to_project_crs_extent(
393 spatial_extent: SpatialExtent, project: QgsProject = None
394) -> typing.Union[QgsRectangle, None]:
395 """Transforms SpatialExtent model to an QGIS extent based
396 on the CRS of the given project.
397
398 :param spatial_extent: Spatial extent data model that defines the
399 scenario bounds.
400 :type spatial_extent: SpatialExtent
401
402 :param project: Project whose CRS will be used to determine
403 the values of the output extent.
404 :type project: QgsProject
405
406 :returns: Output extent in the project's CRS. If the input extent
407 is invalid, this function will return None.
408 :rtype: QgsRectangle
409 """
410
411 input_rect = extent_to_qgs_rectangle(spatial_extent)
412 if input_rect is None:
413 return None
414
415 default_crs = QgsCoordinateReferenceSystem.fromEpsgId(DEFAULT_CRS_I
416 if not default_crs.isValid():
417 return None
418
419 if project is None:
420 project = QgsProject.instance()
421
422 target_crs = project.crs()
423 if default_crs == target_crs:
424

```
# No need for transformation
return input_rect

coordinate_xform = QgsCoordinateTransform(default_crs, project.crs()
return coordinate_xform.transformBoundingBox(input_rect)
```

extent_to_qgs_rectangle

Code:

```
extent_to_qgs_rectangle(spatial_extent)
```

Returns a QgsRectangle object from the SpatialExtent object.

If the SpatialExtent is invalid (i.e. less than four items) then it will return None.

Parameters:

Name	Type	Description	Default
spatial_extent	SpatialExtent	Spatial extent data model that defines the scenario required bounds.	

Returns:

Type	Description
QgsRectangle	QGIS rectangle defining the bounds for the scenario.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/helpers.py

```
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
```

 **Code:**

```
def extent_to_qgs_rectangle(
    spatial_extent: SpatialExtent,
) -> typing.Union[QgsRectangle, None]:
    """Returns a QgsRectangle object from the SpatialExtent object.

    If the SpatialExtent is invalid (i.e. less than four items) then it
    will return None.

    :param spatial_extent: Spatial extent data model that defines the
    scenario bounds.
    :type spatial_extent: SpatialExtent

    :returns: QGIS rectangle defining the bounds for the scenario.
    :rtype: QgsRectangle
    """
    if len(spatial_extent.bbox) < 4:
        return None

    return QgsRectangle(
        spatial_extent.bbox[0],
        spatial_extent.bbox[2],
        spatial_extent.bbox[1],
        spatial_extent.bbox[3],
    )
```

layer_component_to_dict

 **Code:**

```
layer_component_to_dict(layer_component, uuid_to_str=True)
```

Creates a dictionary containing attribute name-value pairs from a layer model component object.

Parameters:

Name	Type	Description	Default
layer_component	LayerModelComponentType	Source layer model component object whose values are to be mapped to the corresponding attribute names.	required
uuid_to_str	bool	Set True to convert the UUID to a string equivalent, else False. Some serialization engines such as JSON are unable to handle UUID objects hence the need to convert to string.	True

Returns:

Type Description

dict	Returns a dictionary item containing attribute name-value pairs.
------	--

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/helpers.py

194

 **Code:**

195

196

197

198

199

200

201

202

203

204

205

206

207

208

209

210

211

212

213

214

215

216

217

218

219

220

221

222

223

224

225

226

227

228

229

230

231

232

233

234

235

236

237

238

239

240

241

242

243

244

245

246

247

248

249

250

251

252

253

254

255

256

257

258

259

260

261

262

263

264

265

266

267

268

269

270

271

272

273

274

275

276

277

278

279

280

281

282

283

284

285

286

287

288

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303

304

305

306

307

308

309

310

311

312

313

314

315

316

317

318

319

320

321

322

323

324

325

326

327

328

329

330

331

332

333

334

335

336

337

338

339

340

341

342

343

344

345

346

347

348

349

350

351

352

353

354

355

356

357

358

359

360

361

362

363

364

365

366

367

368

369

370

371

372

373

374

375

376

377

378

379

380

381

382

383

384

385

386

387

388

389

390

391

392

393

394

395

396

397

398

399

400

401

402

403

404

405

406

407

408

409

410

411

412

413

414

415

416

417

418

419

420

421

422

423

424

425

426

427

428

429

430

431

432

433

434

435

436

437

438

439

440

441

442

443

444

445

446

447

448

449

450

451

452

453

454

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

489

490

491

492

493

494

495

496

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

model_component_to_dict

 **Code:**

```
model_component_to_dict(model_component, uuid_to_str=True)
```

Creates a dictionary containing the base attribute name-value pairs from a model component object.

Parameters:

Name	Type	Description	Default
model_component	BaseModelComponentType	Source model component object whose values are to be mapped to the corresponding attribute names.	required
uuid_to_str	bool	Set True to convert the UUID to a string equivalent, else False. Some serialization engines such as JSON are unable to handle UUID objects hence the need to convert to string.	True

Returns:

Type Description

dict	Returns a dictionary item containing attribute name-value pairs.
------	--

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/helpers.py

 **Code:**

```
41
42
43
44
45
46
47
48 def model_component_to_dict(
49     model_component: BaseModelComponentType, uuid_to_str=True
50 ) -> dict:
51     """Creates a dictionary containing the base attribute
52     name-value pairs from a model component object.
53
54     :param model_component: Source model component object whose
55     values are to be mapped to the corresponding
56     attribute names.
57     :type model_component: BaseModelComponent
58
59     :param uuid_to_str: Set True to convert the UUID to a
60     string equivalent, else False. Some serialization engines
61     such as JSON are unable to handle UUID objects hence the need
62     to convert to string.
63     :type uuid_to_str: bool
64
65
66     :returns: Returns a dictionary item containing attribute
67     name-value pairs.
68     :rtype: dict
69     """
70
71     model_uuid = model_component.uuid
72     if uuid_to_str:
73         model_uuid = str(model_uuid)
74
75     return {
76         UUID_ATTRIBUTE: model_uuid,
77         NAME_ATTRIBUTE: model_component.name,
78         DESCRIPTION_ATTRIBUTE: model_component.description,
79     }
```

ncs_pathway_to_dict

Code:

```
ncs_pathway_to_dict(ncs_pathway, uuid_to_str=True)
```

Creates a dictionary containing attribute name-value pairs from an NCS pathway object.

This function has been retained for legacy support.

Parameters:

Name	Type	Description	Default
ncs_pathway	NcsPathway	Source NCS pathway object whose values are to be mapped required to the corresponding attribute names.	
uuid_to_str	bool	Set True to convert the UUID to a string equivalent, else False. Some serialization engines such as JSON are unable to handle UUID objects hence the need to convert to string.	True

Returns:

Type Description

dict Returns a dictionary item containing attribute name-value pairs.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/models/helpers.py

```
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247
```

 **Code:**

```
def ncs_pathway_to_dict(ncs_pathway: NcsPathway, uuid_to_str=True)  
    """Creates a dictionary containing attribute  
    name-value pairs from an NCS pathway object.  
  
    This function has been retained for legacy support.  
  
    :param ncs_pathway: Source NCS pathway object whose  
    values are to be mapped to the corresponding  
    attribute names.  
    :type ncs_pathway: NcsPathway  
  
    :param uuid_to_str: Set True to convert the UUID to a  
    string equivalent, else False. Some serialization engines  
    such as JSON are unable to handle UUID objects hence the need  
    to convert to string.  
    :type uuid_to_str: bool  
  
    :returns: Returns a dictionary item containing attribute  
    name-value pairs.  
    :rtype: dict  
    """  
  
    base_ncs_dict = layer_component_to_dict(ncs_pathway, uuid_to_st  
    base_ncs_dict[CARBON_PATHS_ATTRIBUTE] = ncs_pathway.carbon_path  
  
    return base_ncs_dict
```

Report

Data models for report production.

ReportContext `dataclass`

Context information for generating a report.

ReportResult `dataclass`

Detailed result information from a report generation run.

pdf_path `property`

 **Code:**

pdf_path

Returns the absolute path to the PDF file if the process completed successfully.

Caller needs to verify if the file actually exists in the given location.

:returns: Absolute path to the PDF file if the process completed successfully else an empty string.

ReportSubmitStatus `dataclass`

Result of report submission process.

1.5.3 GUI

GUI main

The plugin main window class.

QgisCplusMain

 **Code:**

```
QgisCplusMain(iface, parent=None)
```

Bases: [QDockWidget](#), [WidgetUi](#)

Main plugin UI class

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

117
118  **Code:**
119
120
121
122
123
124 def __init__(
125 self,
126 iface,
127 parent=None,
128):
129 super().__init__(parent)
130 self.setupUi(self)
131 self.iface = iface
132 self.progress_dialog = None
133 self.task = None
134 self.processing_cancelled = False
135
136 self.prepare_input()
137
138
139
140
141
142
143
144
145
146 # Insert widget for step 2
147 self.implementation_model_widget = ImplementationModelContainerWidget(
148 self, self.message_bar
149)
150 self.implementation_model_widget.ncs_reloaded.connect(
151 self.on_ncs_pathways_reloaded
152)
153 self.tab_widget.insertTab(
154 1, self.implementation_model_widget, self.tr("Step 2"))
155
156 self.tab_widget.currentChanged.connect(self.on_tab_step_changed)
157
158
159 # Step 3, priority weighting layers initialization
160 self.priority_groups_widgets = {}
161 self.pwl_item_flags = None
162
163
164 self.initialize_priority_layers()

```
    self.position_feedback = QgsProcessingFeedback()
    self.processing_context = QgsProcessingContext()

    self.scenario_result = None

    self.analysis_finished.connect(self.post_analysis)
```

add_priority_group

Code:

```
add_priority_group()
```

Adds a new priority group into the plugin, then updates the priority list to show the new added priority group.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py`

```
657
658  Code:
659
660
661
662
663

def add_priority_group(self):
    """Adds a new priority group into the plugin, then updates
    the priority list to show the new added priority group.
    """
    group_dialog = PriorityGroupDialog()
    group_dialog.exec_()
    self.update_priority_groups()
```

add_priority_layer **Code:**

```
add_priority_layer()
```

Adds a new priority layer into the plugin, then updates the priority list to show the new added priority layer.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py`

```
724  
725  
726  
727  
728  
729  
730
```

```
def add_priority_layer(self):  
    """Adds a new priority layer into the plugin, then updates  
    the priority list to show the new added priority layer.  
    """  
    layer_dialog = PriorityLayerDialog()  
    layer_dialog.exec_()  
    self.update_priority_layers(update_groups=False)
```

add_priority_layer_group **Code:**

```
add_priority_layer_group(  
    target_group=None, priority_layer=None  
)
```

Adds priority layer from the weighting layers into a priority group If no target_group or priority_layer is passed then the current selected group or priority layer from their respective list will be used.

Checks if priority layer is already in the target group and if so no addition is done.

Once the addition is done, the respective priority layer plugin settings are updated to store the new information.

Parameters:

Name	Type	Description	Default
target_group	dict	Priority group where layer will be added to	None
priority_layer	dict	Priority weighting layer to be added	None

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

548
549
550  **Code:**
551
552
553
554
555 def add_priority_layer_group(self, target_group=None, priority_layer=None):
556 """Adds priority layer from the weighting layers into a priority group
557 If no target_group or priority_layer is passed then the current selected
558 group or priority layer from their respective list will be used.
559
560
561
562
563 Checks if priority layer is already in the target group and if so no
564 addition is done.
565
566
567
568 Once the addition is done, the respective priority layer plugin settings
569 are updated to store the new information.
570
571
572 :param target_group: Priority group where layer will be added to
573 :type target_group: dict
574
575
576
577 :param priority_layer: Priority weighting layer to be added
578 :type priority_layer: dict
579
580 """
581
582 selected_priority_layers = (
583 priority_layer or self.priority_layers_list.selectedItems()
584)
585 selected_priority_layers = (
586 [selected_priority_layers]
587 if not isinstance(selected_priority_layers, list)
588 else selected_priority_layers
589)
590
591
592
593
594
595
596 selected_group = target_group or self.priority_groups_list.currentItem()
597
598
599 for selected_priority_layer in selected_priority_layers:
600 if (
601 selected_group is not None and selected_group.parent() is None

```
605 ) and selected_priority_layer is not None:
606     children = selected_group.takeChildren()
607     item_found = False
608     text = selected_priority_layer.data(QtCore.Qt.DisplayRole)
609     for child in children:
610         if child.text(0) == text:
611             item_found = True
612             break
613     selected_group.addChildren(children)
614
615     if not item_found:
616         selected_group.setExpanded(True)
617         item = QtWidgets.QTreeWidgetItem(selected_group)
618         item.setText(0, text)
619         group_widget = self.priority_groups_list.itemWidget(
620             selected_group, 0
621         )
622         layer_id = selected_priority_layer.data(QtCore.Qt.UserRole)
623
624         priority_layer = settings_manager.get_priority_layer(layer_id)
625         item.setData(
626             0,
627             QtCore.Qt.UserRole,
628             priority_layer.get(USER_DEFINED_ATTRIBUTE),
629         )
630         target_group_name = (
631             group_widget.group.get("name") if group_widget.group else
632         )
633
634         groups = priority_layer.get("groups")
635         new_groups = []
636         group_found = False
637
638         for group in groups:
639             if target_group_name == group["name"]:
640                 group_found = True
641
642             if group_found:
```

```
new_group = settings_manager.find_group_by_name(  
    target_group_name  
)  
else:  
    new_group = group  
    new_groups.append(new_group)  
if not group_found:  
    searched_group = settings_manager.find_group_by_name(  
        target_group_name  
)  
    new_groups.append(searched_group)  
  
priority_layer["groups"] = new_groups  
settings_manager.save_priority_layer(priority_layer)  
  
# Trigger check to enable/disable PWLs based on current extent  
self.on_extent_changed(self.extent_box.outputExtent())
```

analysis_complete

 **Code:**

```
analysis_complete(task, report_manager)
```

Calls the responsible function for handling analysis results outputs

Parameters:

Name	Type	Description	Default
task	ScenarioAnalysisTask	Analysis task	required
report_manager	ReportManager	Report manager used to generate analysis reports	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

```
998
999
1000  Code:
1001
1002
1003
1004
1005     def analysis_complete(self, task, report_manager):
1006         """Calls the responsible function for handling analysis results outputs
1007
1008         :param task: Analysis task
1009         :type task: ScenarioAnalysisTask
1010
1011         :param report_manager: Report manager used to generate analysis reports
1012         :type report_manager: ReportManager
1013         """
1014
1015
1016         self.scenario_result = task.scenario_result
1017         self.scenario_results(task, report_manager)
```

cancel_processing_task

 **Code:**

```
cancel_processing_task()
```

Cancels the current processing task.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

1036
1037
1038

 **Code:**

```
def cancel_processing_task(self):
    """Cancels the current processing task."""
    self.processing_cancelled = True
```

edit_priority_group

 **Code:**

```
edit_priority_group()
```

Edits the current selected priority group and updates the group box list.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

665
666
667  **Code:**
668
669
670
671
672
673 def edit_priority_group(self):
674 """Edits the current selected priority group
675 and updates the group box list."""
676 if self.priority_groups_list.currentItem() is None:
677 self.show_message(
678 tr("Select first the priority group from the groups list."),
679 Qgis.Critical,
680)
681 return
682
683 group_identifier = self.priority_groups_list.currentItem().data(
684 0, QtCore.Qt.UserRole
685)
686
687 if group_identifier == "":
688 self.show_message(
689 tr("Could not fetch the selected priority groups for editing."),
690 Qgis.Critical,
691)
692 return
693
694 group = settings_manager.get_priority_group(group_identifier)
695 group_dialog = PriorityGroupDialog(group)
696 group_dialog.exec_()
697 self.update_priority_groups()

edit_priority_layer **Code:**`edit_priority_layer()`

Edits the current selected priority layer and updates the layer box list.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

732
733
734  **Code:**
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753

```
def edit_priority_layer(self):
    """Edits the current selected priority layer
    and updates the layer box list."""
    if self.priority_layers_list.currentItem() is None:
        self.show_message(
            tr("Select first the priority weighting layer from the layers list."),
            Qgis.Critical,
        )
        return

    layer_identifier = self.priority_layers_list.currentItem().data(
        QtCore.Qt.UserRole
    )

    if layer_identifier == "":
        self.show_message(
            tr("Could not fetch the selected priority layer for editing."),
            Qgis.Critical,
        )
        return

    self._show_priority_layer_editor(layer_identifier)
```

group_value_changed

 **Code:**

```
group_value_changed(group_name, group_value)
```

Slot to handle priority group widget changes.

Parameters:

Name	Type	DescriptionDefault
group_name	str	Group name required
group_value	int	Group value required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

449

450

451

452

453

454

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

 **Code:**

```
def group_value_changed(self, group_name, group_value):
    """Slot to handle priority group widget changes.

    :param group_name: Group name
    :type group_name: str

    :param group_value: Group value
    :type group_value: int
    """

group = settings_manager.find_group_by_name(group_name)
group["value"] = group_value
settings_manager.save_priority_group(group)

for index in range(self.priority_groups_list.topLevelItemCount()):
    item = self.priority_groups_list.topLevelItem(index)

    for child_index in range(item.childCount()):
        child = item.child(child_index)
        layer = settings_manager.find_layer_by_name(child.text())
        new_groups = []
        for group in layer.get("groups"):
            if group.get("name") == group_name:
                group["value"] = group_value
            new_groups.append(group)
        layer["groups"] = new_groups
        settings_manager.save_priority_layer(layer)
```

initialize_priority_layers **Code:**

```
initialize_priority_layers()
```

Prepares the priority weighted layers UI with the defaults.

Gets the store priority layers from plugin settings and populates them into the QListWidget as QListWidgetItem s then fetches the priority groups and adds them to the QTreeWidget as QTreeWidgetItem s with their corresponding priority layers as their child items.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

323
324
325  **Code:**
326
327
328
329
330 def initialize_priority_layers(self):
331 """Prepares the priority weighted layers UI with the defaults.
332
333 Gets the store priority layers from plugin settings and populates
334 them into the QListWidget as QListWidgetItem s then fetches the
335 priority groups and adds them to the QTreeWidget as QTreeWidgetItem s
336 with their corresponding priority layers as their child item s.
337 """
338 self.priority_layers_list.clear()
339
340
341
342
343 for layer in settings_manager.get_priority_layers():
344 item = QtWidgets.QListWidgetItem()
345 item.setData(QtCore.Qt.DisplayRole, layer.get("name"))
346 item.setData(QtCore.Qt.UserRole, layer.get("uuid"))
347
348
349 if self.pwl_item_flags is None:
350 self.pwl_item_flags = item.flags()
351
352
353 self.priority_layers_list.addItem(item)
354
355
356
357
358
359
360
361
362 list_items = []
363 items_only = []
364 stored_priority_groups = settings_manager.get_priority_groups()
365 self.priority_groups_list.clear()
366
367
368
369
370 for group in stored_priority_groups:
371 group_widget = PriorityGroupWidget(
372 group,
373)
374 group_widget.input_value_changed.connect(self.group_value_changed)
375 group_widget.slider_value_changed.connect(self.group_value_changed)
376
377
378
379

```
380
381     self.priority_groups_widgets[group["name"]] = group_widget
382
383     pw_layers = settings_manager.find_layers_by_group(group["name"])
384
    item = QtWidgets.QTreeWidgetItem()
    item.setSizeHint(0, group_widget.sizeHint())
    item.setExpanded(True)
    item.setData(0, QtCore.Qt.UserRole, group.get("uuid"))
    # Add priority layers into the group as a child items.
    item.setExpanded(True) if len(pw_layers) > 0 else None
    for layer in pw_layers:
        if item.parent() is None:
            layer_item = QtWidgets.QTreeWidgetItem(item)
            layer_item.setText(0, layer.get("name"))
            layer_item.setData(
                0, QtCore.Qt.UserRole, layer.get(USER_DEFINED_ATTRIBUTE))
        )
        list_items.append((item, group_widget))
        items_only.append(item)
    self.priority_groups_list.addTopLevelItems(items_only)
    for item in list_items:
        self.priority_groups_list.setItemWidget(item[0], 0, item[1])
    # Trigger process to enable/disable PWLs based on current extents
    self.on_extent_changed(self.extent_box.outputExtent())
```

main_task **Code:**`main_task()`

Serves as a QgsTask function for the main task that contains smaller sub-tasks running the actual processing calculations.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py`

1029
1030
1031
1032
1033
1034

```
def main_task(self):
    """Serves as a QgsTask function for the main task that contains
    smaller sub-tasks running the actual processing calculations.
    """

    log("Running from main task.")
```

move_layer_to_group **Code:**`move_layer_to_group(layer, group)`

Moves a layer open in QGIS to another group.

Parameters:

Name	Type	Description	Default
------	------	-------------	---------

layer	QgsRasterLayer	Raster layer to move	required
group	QgsLayerTreeGroup	Group to which the raster should be moved	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

1076

1077

1078

1079

1080

1081

1082

1083

1084

1085

1086

1087

1088

1089

1090

1091

 **Code:**

```
def move_layer_to_group(self, layer, group) -> None:
    """Moves a layer open in QGIS to another group.

    :param layer: Raster layer to move
    :type layer: QgsRasterLayer

    :param group: Group to which the raster should be moved
    :type group: QgsLayerTreeGroup
    """

    if layer:
        instance_root = QgsProject.instance().layerTreeRoot()
        layer = instance_root.findLayer(layer.id())
        layer_clone = layer.clone()
        parent = layer.parent()
        group.insertChildNode(0, layer_clone) # Add to top of g
        parent.removeChildNode(layer)
```

on_extent_changed

 **Code:**

```
on_extent_changed(new_extent)
```

Slot raised when scenario extents have changed.

Used to enable/disable default model items if they are within or outside the pilot AOI.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

 **Code:**

```
393
394
395
396
397
398
399
400     def on_extent_changed(self, new_extent: QgsRectangle):
401         """Slot raised when scenario extents have changed.
402
403         Used to enable/disable default model items if they are within or
404         outside the pilot AOI.
405         """
406
407         within_pilot_area = extent_within_pilot(new_extent, self.extent_box.output
408
409
410         if not within_pilot_area:
411             msg = tr(
412                 "Area of interest is outside the pilot area. Please use your "
413                 "own NCS pathways, implementation models and PWLs."
414             )
415             self.show_message(msg, Qgis.Info)
416
417
418         else:
419             self.message_bar.clearWidgets()
420
421
422         self.implementation_model_widget.enable_default_items(within_pilot_area)
423
424
425         # Enable/disable PWL items
426         for i in range(self.priority_layers_list.count()):
427             pwl_item = self.priority_layers_list.item(i)
428             uuid_str = pwl_item.data(QtCore.Qt.UserRole)
429             if not uuid_str:
430                 continue
431
432             pwl_uuid = uuid.UUID(uuid_str)
433             pwl = settings_manager.get_priority_layer(pwl_uuid)
434             if USER_DEFINED_ATTRIBUTE not in pwl:
435                 continue
```

```
is_user_defined = pwl.get(USER_DEFINED_ATTRIBUTE)
if is_user_defined:
    continue

if within_pilot_area:
    pwl_item.setFlags(self.pwl_item_flags)
else:
    pwl_item.setFlags(QtCore.Qt.NoItemFlags)

# Enable/disable PWL items already defined under the priority groups
for i in range(self.priority_groups_list.topLevelItemCount()):
    group_item = self.priority_groups_list.topLevelItem(i)

    for c in range(group_item.childCount()):
        pwl_tree_item = group_item.child(c)
        is_user_defined = pwl_tree_item.data(0, QtCore.Qt.UserRole)
        if is_user_defined:
            continue

        if within_pilot_area:
            pwl_tree_item.setFlags(self.pwl_item_flags)
        else:
            pwl_tree_item.setFlags(QtCore.Qt.NoItemFlags)
```

on_ncs_pathways_reloaded

 **Code:**

```
on_ncs_pathways_reloaded()
```

Slot raised when NCS pathways have been reloaded in the view.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

386
387
388
389
390
391

 **Code:**

```
def on_ncs_pathways_reloaded(self):
    """Slot raised when NCS pathways have been reloaded in the view."""
    within_pilot_area = extent_within_pilot(
        self.extent_box.outputExtent(), self.extent_box.outputCrs())
    )
    self.implementation_model_widget.enable_default_items(within_pilot_area)
```

on_progress_dialog_cancelled

 **Code:**

```
on_progress_dialog_cancelled()
```

Slot raised when analysis has been cancelled in progress dialog.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

1588
1589
1590
1591

 **Code:**

```
def on_progress_dialog_cancelled(self):
    """Slot raised when analysis has been cancelled in progress dialog."""
    if not self.run_scenario_btn.isEnabled():
        self.run_scenario_btn.setEnabled(True)
```

on_report_finished**Code:**

```
on_report_finished(progress_dialog, scenario_id)
```

Slot raised when report task has finished.

Parameters:

Name	Type	Description	Default
progress_dialog	ProgressDialog	Dialog responsible for showing all the analysis operations progress.	required
scenario_id	str	Scenario analysis id	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

```
1543
1544
1545  Code:
1546
1547
1548
1549
1550     def on_report_finished(self, progress_dialog, scenario_id: str):
1551         """Slot raised when report task has finished.
1552
1553         :param progress_dialog: Dialog responsible for showing
1554             all the analysis operations progress.
1555         :type progress_dialog: ProgressDialog
1556
1557         :param scenario_id: Scenario analysis id
1558         :type scenario_id: str
1559         """
1560
1561         if not self.report_job_is_for_current_scenario(scenario_id):
1562             return
1563
1564         progress_dialog.set_report_complete()
1565         progress_dialog.change_status_message(tr("Report generation complete"))
1566
1567         self.run_scenario_btn.setEnabled(True)
```

on_report_running

```
 Code:
on_report_running(progress_dialog, scenario_id)
```

Slot raised when report task has started.

Parameters:

Name	Type	Description	Default
progress_dialog	ProgressDialog	Dialog responsible for showing all the analysis operations progress.	required
scenario_id	str	Scenario analysis id	required

• Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504

 **Code:**

```
def on_report_running(self, progress_dialog, scenario_id: str):
    """Slot raised when report task has started.

    :param progress_dialog: Dialog responsible for showing
                           all the analysis operations progress.
    :type progress_dialog: ProgressDialog

    :param scenario_id: Scenario analysis id
    :type scenario_id: str
    """

    if not self.report_job_is_for_current_scenario(scenario_id):
        return

    progress_dialog.update_progress_bar(0)
    progress_dialog.report_running = True
    progress_dialog.change_status_message(
        tr("Generating report for the analysis output")
    )
```

```
on_reporting_progress_changed
```

 **Code:**

```
on_reporting_progress_changed(progress_dialog, progress)
```

Slot raised when the reporting progress has changed.

Parameters:

Name	Type	Description	Default
progress_dialog	ProgressDialog	Dialog responsible for showing all the analysis operations progress.	required
progress	float	Analysis progress value between 0 and 100	required

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py`

```
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1540  
1541
```

 **Code:**

```
def on_reporting_progress_changed(self, progress_dialog, progress
    """Slot raised when the reporting progress has changed.

    :param progress_dialog: Dialog responsible for showing
        all the analysis operations progress.
    :type progress_dialog: ProgressDialog

    :param progress: Analysis progress value between 0 and 100
    :type progress: float
    """
    progress_dialog.update_progress_bar(progress)
```

on_tab_step_changed

 **Code:**

```
on_tab_step_changed(index)
```

Slot raised when the current tab changes.

Parameters:

Name	Type	Description	Default
index	int	Zero-based index position of new current tab required	

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

1428
1429
1430 **Code:**
1431
1432
1433
1434
1435 def on_tab_step_changed(self, index: int):
1436 """Slot raised when the current tab changes.
1437
1438 :param index: Zero-based index position of new current tab
1439 :type index: int
1440 """
1441
1442 if index == 1:
1443 self.implementation_model_widget.can_show_error_messages =
1444 self.implementation_model_widget.load()
1445
1446
1447 elif index == 2:
1448 # Validate implementation model selection
1449 selected_implementation_models = (
1450 self.implementation_model_widget.selected_im_items()
1451)
1452 if len(selected_implementation_models) == 0:
1453 msg = self.tr("Please select at least one implementation model")
1454 self.show_message(msg)
1455 self.tab_widget.setCurrentIndex(1)
1456
1457 else:
1458 self.message_bar.clearWidgets()

open_help **Code:**`open_help()`

Opens the user documentation for the plugin in a browser

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py`

653
654
655

 **Code:**

```
def open_help(self):
    """Opens the user documentation for the plugin in a browser"""
    open_documentation(USER_DOCUMENTATION_SITE)
```

open_settings **Code:**`open_settings()`

Options the CPLUS settings in the QGIS options dialog.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

1451
1452
1453

 **Code:**

```
def open_settings(self):
    """Options the CPLUS settings in the QGIS options dialog."""
    self.iface.showOptionsDialog(currentPage=OPTIONS_TITLE)
```

post_analysis **Code:**

```
post_analysis(scenario_result, task, report_manager)
```

Handles analysis outputs from the final analysis results. Adds the resulting scenario raster to the canvas with styling. Adds each of the implementation models to the canvas with styling. Adds each IMs pathways to the canvas.

Parameters:

Name	Type	Description	Default
scenario_result	ScenarioResult	ScenarioResult of output results	required
task	ScenarioAnalysisTask	Analysis task	required
report_manager	ReportManager	Report manager used to generate analysis reports	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

 **Code:**

```
1093
1094
1095
1096
1097
1098
1099
1100 def post_analysis(self, scenario_result, task, report_manager):
1101     """Handles analysis outputs from the final analysis results.
1102     Adds the resulting scenario raster to the canvas with styling.
1103     Adds each of the implementation models to the canvas with styling.
1104     Adds each IMs pathways to the canvas.
1105
1106     :param scenario_result: ScenarioResult of output results
1107     :type scenario_result: ScenarioResult
1108
1109     :param task: Analysis task
1110     :type task: ScenarioAnalysisTask
1111
1112
1113     :param report_manager: Report manager used to generate analysis reports
1114     :type report_manager: ReportManager
1115     """
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125 # If the processing were stopped, no file will be added
1126 if not self.processing_cancelled:
1127     list_models = scenario_result.scenario.models
1128     raster = scenario_result.analysis_output["OUTPUT"]
1129     im_weighted_dir = os.path.join(os.path.dirname(raster), "weighted_ims")
1130
1131
1132
1133
1134
1135     scenario_name = scenario_result.scenario.name
1136     qgis_instance = QgsProject.instance()
1137     instance_root = qgis_instance.layerTreeRoot()
1138
1139
1140
1141     # Check if there are other groups for the scenario
1142     # and assign a suffix.
1143     counter = 1
1144     group_name = scenario_name
1145
1146
1147
1148
1149
```

```
1150     # Control to prevent infinite loop
1151     max_limit = 100
1152     while True and counter <= max_limit:
1153         scenario_grp = instance_root.findGroup(group_name)
1154         if scenario_grp is None:
1155             break
1156         group_name = f'{scenario_name} {counter}s}'
1157         counter += 1
1158
1159     # Groups
1160     scenario_group = instance_root.insertGroup(0, group_name)
1161     im_group = scenario_group.addGroup(tr(IM_GROUP_LAYER_NAME))
1162     im_weighted_group = (
1163         scenario_group.addGroup(tr(IM_WEIGHTED_GROUP_NAME))
1164         if os.path.exists(im_weighted_dir)
1165         else None
1166     )
1167     pathways_group = scenario_group.addGroup(tr(NCS_PATHWAYS_GROUP_LAYER_NAME))
1168
1169     # Group settings
1170     im_group.setExpanded(False)
1171     im_weighted_group.setExpanded(False) if im_weighted_group else None
1172     pathways_group.setExpanded(False)
1173     pathways_group.setItemVisibilityCheckedRecursive(False)
1174
1175
1176     # Add scenario result layer to the canvas with styling
1177     layer_file = scenario_result.analysis_output.get("OUTPUT")
1178     layer_name = (
1179         f'{SCENARIO_OUTPUT_LAYER_NAME}_'
1180         f'{datetime.datetime.now().strftime("%Y_%m_%d_%H_%M_%S")}'
1181     )
1182     scenario_result.output_layer_name = layer_name
1183     layer = QgsRasterLayer(layer_file, layer_name, QGIS_GDAL_PROVIDER)
1184     scenario_layer = qgis_instance.addMapLayer(layer)
1185
1186     # Scenario result layer styling
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
```

```
1207     renderer = self.style_models_layer(layer, self.analysis_weighted_ims)
1208     layer.setRenderer(renderer)
1209     layer.triggerRepaint()
1210
1211
1212     """A workaround to add a layer to a group.
1213     Adding it using group.insertChildNode or group.addLayer causes issues,
1214     but adding to the root is fine.
1215     This approach adds it to the root, and then moves it to the group.
1216     """
1217
1218     self.move_layer_to_group(scenario_layer, scenario_group)
1219
1220
1221
1222     # Add implementation models and pathways
1223     im_index = 0
1224     for im in list_models:
1225         im_name = im.name
1226         im_layer = QgsRasterLayer(im.path, im.name)
1227         im_layer.setCustomProperty(MODEL_IDENTIFIER_PROPERTY, str(im.uuid))
1228         list_pathways = im.pathways
1229
1230
1231         # Add IM layer with styling, if available
1232         if im_layer:
1233             renderer = self.style_model_layer(im_layer, im)
1234
1235
1236             added_im_layer = qgis_instance.addMapLayer(im_layer)
1237             self.move_layer_to_group(added_im_layer, im_group)
1238
1239
1240             im_layer.setRenderer(renderer)
1241             im_layer.triggerRepaint()
1242
1243
1244         # Add IM pathways
1245         if len(list_pathways) > 0:
1246             # im_pathway_group = pathways_group.addGroup(im_name)
1247             im_pathway_group = pathways_group.insertGroup(im_index, im_name)
1248             im_pathway_group.setExpanded(False)
1249
1250
1251
1252         pw_index = 0
1253
1254
1255
1256
```

```
for pathway in list_pathways:
    try:
        # pathway_name = pathway.name
        pathway_layer = pathway.to_map_layer()

        added_pw_layer = qgis_instance.addMapLayer(pathway_layer)
        self.move_layer_to_group(added_pw_layer, im_pathway_group)

        pathway_layer.triggerRepaint()

        pw_index = pw_index + 1
    except Exception as err:
        self.show_message(
            tr(
                "An error occurred loading a pathway, "
                "check logs for more information"
            ),
            level=Qgis.Info,
        )
        log(
            tr(
                "An error occurred loading a pathway, "
                'scenario analysis, error message "{}"'.format(
                    err
                )
            )
        )

    im_index = im_index + 1

weighted_ims = task.analysis_weighted_ims if task is not None else []

for model in weighted_ims:
    weighted_im_path = model.path
    weighted_im_name = Path(weighted_im_path).stem

    if not weighted_im_path.endswith(".tif"):
        continue
```

```
im_weighted_layer = QgsRasterLayer(  
    weighted_im_path, weighted_im_name, QGIS_GDAL_PROVIDER  
)  
  
# Set UUID for easier retrieval  
im_weighted_layer.setCustomProperty(  
    MODEL_IDENTIFIER_PROPERTY, str(model.uuid)  
)  
  
renderer = self.style_model_layer(im_weighted_layer, model)  
im_weighted_layer.setRenderer(renderer)  
im_weighted_layer.triggerRepaint()  
  
added_im_weighted_layer = qgis_instance.addMapLayer(im_weighted_layer)  
self.move_layer_to_group(added_im_weighted_layer, im_weighted_group)  
  
# Initiate report generation  
self.run_report(task.progress_dialog, report_manager)  
else:  
    # Reinitializes variables if processing were cancelled by the user  
    # Not doing this breaks the processing if a user tries to run  
    # the processing after cancelling or if the processing fails  
    self.position_feedback = QgsProcessingFeedback()  
    self.processing_context = QgsProcessingContext()
```

prepare_extent_box

Code:

```
prepare_extent_box()
```

Configure the spatial extent box with the initial settings.

- Source code in `/home/samwel/i/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py`

```
1406
1407  Code:
1408
1409
1410
1411
1412
1413 def prepare_extent_box(self):
1414     """Configure the spatial extent box with the initial settings."""
1415
1416
1417     self.extent_box.setOutputCrs(QgsCoordinateReferenceSystem("EPSG:4326"))
1418
1419     map_canvas = iface.mapCanvas()
1420
1421     self.extent_box.setCurrentExtent(
1422         map_canvas.mapSettings().destinationCrs().bounds(),
1423         map_canvas.mapSettings().destinationCrs(),
1424
1425     )
1426
1427     self.extent_box.setOutputExtentFromCurrent()
1428
1429     self.extent_box.setMapCanvas(map_canvas)
1430
1431
1432     extent_list = PILOT_AREA_EXTENT["coordinates"]
1433     default_extent = QgsRectangle(
1434         extent_list[0], extent_list[2], extent_list[1], extent_list[3]
1435     )
1436
1437
1438     self.extent_box.setOutputExtentFromUser(
1439         default_extent,
1440         QgsCoordinateReferenceSystem("EPSG:4326"),
1441
1442     )
```

prepare_input

 **Code:**

```
prepare_input()
```

Initializes plugin input widgets

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

156
157  **Code:**
158
159
160
161
162
163 def prepare_input(self):
164 """Initializes plugin input widgets"""
165 self.prepare_extent_box()
166 self.grid_layout = QtWidgets.QGridLayout()
167 self.message_bar = QgsMessageBar()
168 self.prepare_message_bar()
169
170
171
172
173
174 self.progress_dialog = None
175 self.scenario_directory = None
176
177
178
179 self.help_btn.clicked.connect(self.open_help)
180 self.pilot_area_btn.clicked.connect(self.zoom_pilot_area)
181
182
183
184 self.run_scenario_btn.clicked.connect(self.run_analysis)
185 self.options_btn.clicked.connect(self.open_settings)
186
187
188 self.restore_scenario()
189
190
191
192 self.scenario_name.textChanged.connect(self.save_scenario)
193 self.scenario_description.textChanged.connect(self.save_scenario)
194 self.extent_box.extentChanged.connect(self.save_scenario)
195
196
197 # Monitors if current extents are within the pilot AOI
198 self.extent_box.extentChanged.connect(self.on_extent_changed)
199
200
201
202
203 icon_pixmap = QtGui.QPixmap(ICON_PATH)
204 self.icon_la.setPixmap(icon_pixmap)
205
206
207
208 add_layer_icon = QtGui.QIcon(ADD_LAYER_ICON_PATH)
209 self.layer_add_btn.setIcon(add_layer_icon)
210
211
212

```
213     remove_layer_icon = QtGui.QIcon(REMOVE_LAYER_ICON_PATH)
214     self.layer_remove_btn.setIcon(remove_layer_icon)
215
216
217     self.layer_add_btn.clicked.connect(self.add_priority_layer_group)
218     self.layer_remove_btn.clicked.connect(self.remove_priority_layer_group)
219
220
221     # Priority groups buttons
222     self.add_group_btn.setIcon(FileUtils.get_icon("symbologyAdd.svg"))
223     self.edit_group_btn.setIcon(FileUtils.get_icon("mActionToggleEditing.svg"))
224     self.remove_group_btn.setIcon(FileUtils.get_icon("symbologyRemove.svg"))
225
226
227     self.add_group_btn.clicked.connect(self.add_priority_group)
228     self.edit_group_btn.clicked.connect(self.edit_priority_group)
229     self.remove_group_btn.clicked.connect(self.remove_priority_group)
230
231
232     # Priority layers buttons
233     self.add_pwl_btn.setIcon(FileUtils.get_icon("symbologyAdd.svg"))
234     self.edit_pwl_btn.setIcon(FileUtils.get_icon("mActionToggleEditing.svg"))
235     self.remove_pwl_btn.setIcon(FileUtils.get_icon("symbologyRemove.svg"))
236
237
238     self.add_pwl_btn.clicked.connect(self.add_priority_layer)
239     self.edit_pwl_btn.clicked.connect(self.edit_priority_layer)
240     self.remove_pwl_btn.clicked.connect(self.remove_priority_layer)
241
242
243
244
245     self.priority_layers_list.itemDoubleClicked.connect(
246         self._on_double_click_priority_layer
247     )
248
249
250     # Add priority groups list into the groups frame
251     self.priority_groups_list = CustomTreeWidget()
252
253
254     self.priority_groups_list.setHeaderHidden(True)
255
256
257     self.priority_groups_list.setDragEnabled(True)
258     self.priority_groups_list.setDragDropOverwriteMode(True)
259     self.priority_groups_list.viewport().setAcceptDrops(True)
```

```
self.priority_groups_list.setDragDropMode(QtWidgets.QAbstractItemView.DragOnly)

self.priority_groups_list.child_dragged_dropped.connect(
    self.priority_groups_update
)

layout = QtWidgets.QVBoxLayout()
layout.setSpacing(0)
layout.setContentsMargins(0, 0, 0, 0)

layout.addWidget(self.priority_groups_list)
self.priority_groups_frame.setLayout(layout)

# Scenario analysis variables

self.analysis_scenario_name = None
self.analysis_scenario_description = None
self.analysis_extent = None
self.analysis_implementation_models = None
self.analysis_weighted_ims = []
self.analysis_priority_layers_groups = []
```

prepare_message_bar

```
prepare_message_bar()
```

Initializes the widget message bar settings

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

```
798  
799  
800 801  
802  
803  
804  
805 806
```

 **Code:**

```
def prepare_message_bar(self):  
    """Initializes the widget message bar settings"""  
    self.message_bar.setSizePolicy(  
        QtWidgets.QSizePolicy.Minimum, QtWidgets.QSizePolicy.Fixed  
    )  
    self.grid_layout.addWidget(  
        self.message_bar, 0, 0, 1, 1, alignment=QtCore.Qt.AlignTop  
    )  
    self.dock_widget_contents.layout().insertLayout(0, self.grid_
```

priority_groups_update

 **Code:**

```
priority_groups_update(target_item, selected_items)
```

Updates the priority groups list item with the passed selected layer items.

Parameters:

Name	Type	Description	Default
target_item	QTreeWidgetItem	The priority group tree widget item that is to be required updated	
selected_items	list	Priority layers items from the list widget	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

```
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260
```

 **Code:**

```
def priority_groups_update(self, target_item, selected_items):  
    """Updates the priority groups list item with the passed  
    selected layer items.  
  
    :param target_item: The priority group tree widget  
    item that is to be updated  
    :type target_item: QTreeWidgetItem  
  
    :param selected_items: Priority layers items from the list widget  
    :type selected_items: list  
    """  
    self.priority_groups_list.setCurrentItem(target_item)  
  
    for item in selected_items:  
        self.add_priority_layer_group(target_item, item)
```

remove_priority_group

 **Code:**

```
remove_priority_group()
```

Removes the current active priority group.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py`

 **Code:**

```
691
692
693
694
695
696
697
698 def remove_priority_group(self):
699     """Removes the current active priority group."""
700     if self.priority_groups_list.currentItem() is None:
701         self.show_message(
702             tr("Select first the priority group from the groups list"),
703             Qgis.Critical,
704         )
705         return
706     group_identifier = self.priority_groups_list.currentItem().data(
707         0, QtCore.Qt.UserRole
708     )
709
710     group = settings_manager.get_priority_group(group_identifier)
711     current_text = group.get("name")
712
713     if group_identifier is None or group_identifier == "":
714         self.show_message(
715             tr("Could not fetch the selected priority group for editing."),
716             Qgis.Critical,
717         )
718         return
719
720
721     reply = QtWidgets.QMessageBox.warning(
722         self,
723         tr("QGIS CPLUS PLUGIN"),
724         tr('Remove the priority group "{}"?').format(current_text),
725         QtWidgets.QMessageBox.Yes,
726         QtWidgets.QMessageBox.No,
727     )
728     if reply == QtWidgets.QMessageBox.Yes:
```

```
    settings_manager.delete_priority_group(group_identifier)
    self.update_priority_groups()
```

remove_priority_layer

 **Code:**

```
remove_priority_layer()
```

Removes the current active priority layer.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

767
768
769  **Code:**
770
771
772
773
774
775 def remove_priority_layer(self):
776 """Removes the current active priority layer."""
777 if self.priority_layers_list.currentItem() is None:
778 self.show_message(
779 tr(
780 >Select first the priority "weighting layer from the layers
781),
782 Qgis.Critical,
783)
784 return
785 current_text = self.priority_layers_list.currentItem().data(
786 QtCore.Qt.DisplayRole
787)
788 if current_text == "":
789 self.show_message(
790 tr("Could not fetch the selected priority layer for editing."),
791 Qgis.Critical,
792)
793 return
794 layer = settings_manager.find_layer_by_name(current_text)
795 reply = QtWidgets.QMessageBox.warning(
796 self,
797 tr("QGIS CPLUS PLUGIN"),
798 tr('Remove the priority layer "{}"?').format(current_text),
799 QtWidgets.QMessageBox.Yes,
800 QtWidgets.QMessageBox.No,
801)

802 if reply == QtWidgets.QMessageBox.Yes:
803 settings_manager.delete_priority_layer(layer.get("uuid"))
804 self.update_priority_layers(update_groups=False)

remove_priority_layer_group

Code:

```
remove_priority_layer_group()
```

Remove the current select priority layer from the current priority group.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

```
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651
```

Code:

```
def remove_priority_layer_group(self):  
    """Remove the current select priority layer from the current priority group  
    selected_group = self.priority_groups_list.currentItem()  
    parent_item = selected_group.parent() if selected_group is not None else None  
  
    if parent_item:  
        priority_layer = settings_manager.find_layer_by_name(selected_group.text())  
        group_widget = self.priority_groups_list.itemWidget(parent_item, 0)  
  
        groups = priority_layer.get("groups")  
        new_groups = []  
        for group in groups:  
            if group.get("name") == group_widget.group.get("name"):  
                continue  
            new_groups.append(group)  
        priority_layer["groups"] = new_groups  
        settings_manager.save_priority_layer(priority_layer)  
  
        parent_item.removeChild(selected_group)
```

report_job_is_for_current_scenario

Code:

```
report_job_is_for_current_scenario(scenario_id)
```

Checks if the given scenario identifier is for the current scenario result.

This is to ensure that signals raised by the report manager refer to the current scenario result object and not for old jobs.

Parameters:

Name	Type	Description	Default
scenario_id	str	Scenario identifier usually from a signal raised by the report manager.required	

Returns:

Type Description

bool True if the scenario identifier matches the current scenario object in the results, else False.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

1561
1562  **Code:**
1563
1564
1565
1566
1567
1568 def report_job_is_for_current_scenario(self, scenario_id: str) -> bool:
1569 """Checks if the given scenario identifier is for the current
1570 scenario result.
1571
1572 This is to ensure that signals raised by the report manager refer
1573 to the current scenario result object and not for old jobs.
1574
1575 :param scenario_id: Scenario identifier usually from a signal
1576 raised by the report manager.
1577 :type scenario_id: str
1578
1579 :returns: True if the scenario identifier matches the current
1580 scenario object in the results, else False.
1581 :rtype: bool
1582 """
1583 if self.scenario_result is None:
1584 return False
1585
1586 current_scenario = self.scenario_result.scenario
1587 if current_scenario is None:
1588 return False
1589
1590 if str(current_scenario.uuid) == scenario_id:
1591 return True
1592
1593 return False

```
reset_reporting_feedback

Code:

reset_reporting_feedback(progress_dialog)
```

Creates a new reporting feedback object and reconnects the signals.

We are doing this to address cases where the feedback is canceled and the same object has to be reused for subsequent report generation tasks.

Parameters:

Name	Type	Description	Default
progress_dialog	ProgressDialog	Dialog responsible for showing all the analysis required operations progress.	

Returns:

Type	Description
QgsFeedback	Feedback instance to be used in storing processing status details.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

1506
1507  **Code:**
1508
1509
1510
1511
1512
1513 def reset_reporting_feedback(self, progress_dialog):
1514 """Creates a new reporting feedback object and reconnects
1515 the signals.
1516
1517 We are doing this to address cases where the feedback is canceled
1518 and the same object has to be reused for subsequent report
1519 generation tasks.
1520
1521
1522
1523
1524
1525
1526 :param progress_dialog: Dialog responsible for showing
1527 all the analysis operations progress.
1528 :type progress_dialog: ProgressDialog
1529
1530 :returns reporting_feedback: Feedback instance to be used in storing
1531 processing status details.
1532 :rtype reporting_feedback: QgsFeedback
1533
1534 """
1535
1536 progress_changed = partial(self.on_reporting_progress_changed, progress_dialog)
1537
1538 reporting_feedback = QgsFeedback(self)
1539 reporting_feedback.progressChanged.connect(progress_changed)
1540
1541 return reporting_feedback

restore_scenario



Code :

`restore_scenario()`

Update the first tab input with the last scenario details

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qqgis_cplus_main.py`

303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321



Code:

```
def restore_scenario(self):
```

"""\Update the first tab input with the last scenario details""""

```
scenario_name = settings_manager.get_value(Settings.SCENARIO_NAME)
```

scenario_description = settings_manager.get_value(Settings SCENAE)

```
extent = settings_manager.get_value(Settings.SCENARIO_EXTENT)
```

```
EXTENT = settings_manager.get_value(settings.SCENARIO_EXTENT)
```

```
self.scenario_name.setText(scenario_name) if scenario_name is
```

```
self.scenario_description.set_text(
```

scenario_description

) if scenario_description is not None else None

if extent is not None:

```
extent_rectangle = QgsRectangle(
```

```
float(extent[0]), float(extent[2]), float(extent[1]), flc
```

)

```
self.extent_box.setOutputExtentFromUser()
```

extent rectangle.

```
OgcCoordinateReferenceSystem("EPSG:4326")
```

1

run_analysis**Code:**

```
run_analysis()
```

Runs the plugin analysis Creates new QgsTask, progress dialog and report manager for each new scenario analysis.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

808
809
810  **Code:**
811
812
813
814
815 def run_analysis(self):
816 """Runs the plugin analysis
817 Creates new QgsTask, progress dialog and report manager
818 for each new scenario analysis.
819 """
820
821
822
823
824
825 extent_list = PILOT_AREA_EXTENT["coordinates"]
826 default_extent = QgsRectangle(
827 extent_list[0], extent_list[2], extent_list[1], extent_list[3]
828)
829
830
831 passed_extent = self.extent_box.outputExtent()
832 contains = default_extent == passed_extent or default_extent.contains(
833 passed_extent
834)
835
836 self.analysis_scenario_name = self.scenario_name.text()
837 self.analysis_scenario_description = self.scenario_description.text()
838
839
840
841
842 self.position_feedback = QgsProcessingFeedback()
843 self.processing_context = QgsProcessingContext()
844
845
846
847 for group in settings_manager.get_priority_groups():
848 group_layer_dict = {
849 "name": group.get("name"),
850 "value": group.get("value"),
851 "layers": [],
852 }
853
854 for layer in settings_manager.get_priority_layers():
855 pwl_items = self.priority_layers_list.findItems(
856 layer.get("name"), QtCore.Qt.MatchExactly
857)
858 if len(pwl_items) > 0:
859
860
861
862
863
864

```
865         # Exclude adding the PWL since its for a disabled default
866         # item outside the pilot AOI.
867         if pwl_items[0].flags() == QtCore.Qt.NoItemFlags:
868             continue
869
870
871
872         group_names = [group.get("name") for group in layer.get("groups",
873             if group.get("name") in group_names:
874                 group_layer_dict["layers"].append(layer.get("name"))
875                 self.analysis_priority_layers_groups.append(group_layer_dict)
876
877
878
879         self.analysis_implementation_models = [
880             item.implementation_model
881             for item in self.implementation_model_widget.selected_im_items()
882             if item.isEnabled()
883
884         ]
885
886
887
888
889         self.analysis_weighted_ims = []
890
891
892
893         base_dir = settings_manager.get_value(Settings.BASE_DIR)
894
895
896         if self.analysis_scenario_name == "" or self.analysis_scenario_name is None:
897             self.show_message(
898                 tr(f"Scenario name cannot be blank."),
899                 level=Qgis.Critical,
900                 )
901             return
902
903
904
905         if (
906             self.analysis_scenario_description == ""
907             or self.analysis_scenario_description is None
908         ):
909             self.show_message(
910                 tr(f"Scenario description cannot be blank."),
911                 level=Qgis.Critical,
912                 )
913             return
914
915
916
917
918
919
920
921
```

```
922         self.analysis_implementation_models == []
923     or self.analysis_implementation_models is None
924     ):
925         self.show_message(
926             tr("Select at least one implementation models from step two."),
927             level=Qgis.Critical,
928         )
929         return
930
931
932
933
934
935
936     if not contains:
937         self.show_message(
938             tr(f"Selected area of interest is outside the pilot area."),
939             level=Qgis.Info,
940         )
941         default_ext = (
942             f"{default_extent.xMinimum()}, {default_extent.xMaximum()},"
943             f"{default_extent.yMinimum()}, {default_extent.yMaximum()}"
944         )
945         log(
946             f"Outside the pilot area, passed extent "
947             f"{passed_extent}"
948             f"default extent{default_ext}"
949         )
950
951
952
953
954
955
956
957
958
959     if base_dir is None:
960         self.show_message(
961             tr(
962                 f"Plugin base data directory is not set! "
963                 f"Go to plugin settings in order to set it."
964             ),
965             level=Qgis.Critical,
966         )
967
968
969
970
971
972
973     self.analysis_extent = SpatialExtent(
974         bbox=[
975             passed_extent.xMinimum(),
976             passed_extent.xMaximum(),
977             passed_extent.yMinimum(),
978             passed_extent.yMaximum()
```



```
analysis_task = ScenarioAnalysisTask(  
    self.analysis_scenario_name,  
    self.analysis_scenario_description,  
    self.analysis_implementation_models,  
    self.analysis_priority_layers_groups,  
    self.analysis_extent,  
    scenario,  
    self,  
    progress_dialog,  
)  
  
progress_dialog.analysis_task = analysis_task  
  
report_running = partial(self.on_report_running, progress_dialog)  
report_finished = partial(self.on_report_finished, progress_dialog)  
  
# Report manager  
scenario_report_manager = report_manager  
  
scenario_report_manager.generate_started.connect(report_running)  
scenario_report_manager.generate_completed.connect(report_finished)  
  
analysis_complete = partial(  
    self.analysis_complete, analysis_task, scenario_report_manager  
)  
  
analysis_task.taskCompleted.connect(analysis_complete)  
  
analysis_task.taskTerminated.connect(self.task_terminated)  
  
QgsApplication.taskManager().addTask(analysis_task)  
  
except Exception as err:  
    self.show_message(  
        tr("An error occurred when preparing analysis task"),
```

```
        level=Qgis.Info,
    )
    log(
        tr(
            "An error occurred when preparing analysis task"
            ', error message "{}"'.format(err)
        )
    )
)
```

run_report

Code:

```
run_report(progress_dialog, report_manager)
```

Run report generation. This should be called after the analysis is complete.

Parameters:

Name	Type	Description	Default
progress_dialog	ProgressDialog	Dialog responsible for showing all the analysis operations progress.	required
report_manager	ReportManager	Report manager used to generate analysis reports	required

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py`

1455
1456
1457  **Code:**
1458
1459
1460
1461
1462 def run_report(self, progress_dialog, report_manager):
1463 """Run report generation. This should be called after the
1464 analysis is complete.
1465
1466 :param progress_dialog: Dialog responsible for showing
1467 all the analysis operations progress.
1468 :type progress_dialog: ProgressDialog
1469
1470 :param report_manager: Report manager used to generate analysis reports
1471 :type report_manager: ReportManager
1472 """
1473
1474 if self.processing_cancelled:
1475 # Will not proceed if processing has been cancelled by the user
1476 return
1477
1478 if self.scenario_result is None:
1479 log(
1480 "Cannot run report generation, scenario result is not defined",
1481 info=False,
1482)
1483 return
1484
1485
1486 reporting_feedback = self.reset_reporting_feedback(progress_dialog)
1487 self.reporting_feedback = reporting_feedback
1488
1489 submit_result = report_manager.generate(
1490 self.scenario_result, reporting_feedback
1491)
1492 if not submit_result.status:
1493 msg = self.tr("Unable to submit report request for scenario")
1494 self.show_message(f"{{msg}} {{self.scenario_result.scenario.name}}.")

save_scenario **Code:**`save_scenario()`

Save current scenario details into settings

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py`

286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301

 **Code:**

```
def save_scenario(self):
    """Save current scenario details into settings"""
    scenario_name = self.scenario_name.text()
    scenario_description = self.scenario_description.text()
    extent = self.extent_box.outputExtent()

    extent_box = [
        extent.xMinimum(),
        extent.xMaximum(),
        extent.yMinimum(),
        extent.yMaximum(),
    ]

    settings_manager.set_value(Settings.SCENARIO_NAME, scenario_name)
    settings_manager.set_value(Settings.SCENARIO_DESCRIPTION, scenario_description)
    settings_manager.set_value(Settings.SCENARIO_EXTENT, extent_box)
```

```
scenario_results

Code:

scenario_results(task, report_manager)
```

Called when the task ends. Sets the progress bar to 100 if it finished.

Parameters:

Name	Type	Description	Default
task	ScenarioAnalysisTask	Analysis task	required
report_manager	ReportManager	Report manager used to generate analysis reports	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

```
1056
1057
1058  Code:
1059
1060
1061
1062
1063     def scenario_results(self, task, report_manager):
1064         """Called when the task ends. Sets the progress bar to 100 if it finished.
1065
1066         :param task: Analysis task
1067         :type task: ScenarioAnalysisTask
1068
1069         :param report_manager: Report manager used to generate analysis reports
1070         :type report_manager: ReportManager
1071
1072         """
1073
1074         if task.output is not None:
1075             self.update_progress_bar(task.progress_dialog, 100)
1076             self.scenario_result.analysis_output = task.output
1077             self.scenario_result.state = ScenarioState.FINISHED
1078             self.post_analysis(self.scenario_result, task, report_manager)
1079         else:
1080             task.progress_dialog.change_status_message(
1081                 "No valid output from the processing results."
1082             )
1083             log(f"No valid output from the processing results.")
```

show_message

 **Code:**

```
show_message(message, level=Qgis.Warning)
```

Shows message on the main widget message bar.

Parameters:

Name	Type	Description	Default
message	str	Text message	required
level	Qgis.MessageLevel	Message level type	Warning

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

1360

1361

1362

1363

1364

1365

1366

1367

1368

1369

1370

 **Code:**

```

def show_message(self, message, level=Qgis.Warning):
    """Shows message on the main widget message bar.

    :param message: Text message
    :type message: str

    :param level: Message level type
    :type level: Qgis.MessageLevel
    """

    self.message_bar.clearWidgets()
    self.message_bar.pushMessage(message, level=level)

```

style_model_layer

 **Code:**

```
style_model_layer(layer, model)
```

Applies the styling to the layer that contains the passed implementation model name.

Parameters:

Name	Type	Description	Default
layer	QgsRasterLayer	Raster layer to which to apply the symbology	required
model	ImplementationModel	Implementation model	required

Returns:

Type	Description
<code>QgsSingleBandPseudoColorRenderer</code>	Renderer for the symbology.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

1288
1289  **Code:**

1290
1291
1292
1293
1294
1295 def style_model_layer(self, layer, model):
1296 """Applies the styling to the layer that contains the passed
1297 implementation model name.
1298
1299 :param layer: Raster layer to which to apply the symbology
1300 :type layer: QgsRasterLayer
1301
1302 :param model: Implementation model
1303 :type model: ImplementationModel
1304
1305 :returns: Renderer for the symbology.
1306 :rtype: QgsSingleBandPseudoColorRenderer
1307 """
1308
1309
1310
1311
1312
1313
1314
1315

```
# Retrieves a build-in QGIS color ramp
color_ramp = model.model_color_ramp()

stats = layer.dataProvider().bandStatistics(1)
renderer = QgsSingleBandPseudoColorRenderer(layer.dataProvider)

renderer.setClassificationMin(stats.minimumValue)
renderer.setClassificationMax(stats.maximumValue)

renderer.createShader(
    color_ramp, QgsColorRampShader.Interpolated, QgsColorRampS
)

return renderer
```

```
style_models_layer
```

 **Code:**

```
style_models_layer(layer, models)
```

Applies the styling to the passed layer that contains the passed list of models.

Parameters:

Name	Type	Description	Default
layer	QgsRasterLayer	Layer to be styled	required
models	list	List which contains the implementation models that were passed to the highest position analysis tool	required

Returns:

Type	Description
QgsPalettesRasterRenderer	Renderer for the symbology.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

1258
1259
1260  **Code:**
1261
1262
1263
1264
1265 def style_models_layer(self, layer, models):
1266 """Applies the styling to the passed layer that
1267 contains the passed list of models.
1268
1269 :param layer: Layer to be styled
1270 :type layer: QgsRasterLayer
1271
1272 :param models: List which contains the implementation
1273 models that were passed to the highest position analysis tool
1274 :type models: list
1275
1276 :returns: Renderer for the symbology.
1277 :rtype: QgsPalettesRasterRenderer
1278 """
1279
1280 area_classes = []
1281 for model in models:
1282 im_name = model.name
1283
1284 raster_val = model.style_pixel_value
1285 color = model.scenario_fill_symbol().color()
1286 color_ramp_shader = QgsColorRampShader.ColorRampItem(
1287 float(raster_val), QtGui.QColor(color), im_name
1288)
1289 area_classes.append(color_ramp_shader)
1290
1291 class_data = QgsPalettesRasterRenderer.colorTableToClassData(ar
1292 renderer = QgsPalettesRasterRenderer(layer.dataProvider(), 1, c
1293
1294 return renderer

task_terminated **Code:**

```
task_terminated()
```

Handles logging of the scenario analysis task status after it has been terminated.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

992
993
994
995
996

```
def task_terminated(self):  
    """Handles logging of the scenario analysis task status  
    after it has been terminated.  
    """  
    log(f"Main task terminated")
```

transform_extent **Code:**

```
transform_extent(extent, source_crs, dest_crs)
```

Transforms the passed extent into the destination crs

:param extent: Target extent

Parameters:

Name	Type	Description	Default
source_crs	QgsCoordinateReferenceSystem	Source CRS of the passed extent	required
dest_crs	QgsCoordinateReferenceSystem	Destination CRS	required

- Source code in `/home/samwelij/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py`

```
1011  
1012  Code:  
1013  
1014  
1015  
1016  
1017  
1018 def transform_extent(self, extent, source_crs, dest_crs):  
1019     """Transforms the passed extent into the destination crs  
1020  
1021         :param extent: Target extent  
1022         :type extent: QgsRectangle  
1023  
1024         :param source_crs: Source CRS of the passed extent  
1025         :type source_crs: QgsCoordinateReferenceSystem  
1026  
1027         :param dest_crs: Destination CRS  
1028         :type dest_crs: QgsCoordinateReferenceSystem  
1029     """  
1030  
1031     transform = QgsCoordinateTransform(source_crs, dest_crs, QgsCoordinateTransform.QgsCoordinateTransform.  
1032                                         Transformer.CrsTransform)  
1033     transformed_extent = transform.transformBoundingBox(extent)  
1034  
1035     return transformed_extent
```

update message bar

Code :

`update_message_bar(message)`

Changes the message in the message bar item.

Parameters:

Name	Type	Description	Default
------	------	-------------	---------

message str Message to be updated required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

1351
1352 # Code:
1353
1354
1355
1356
1357
1358

```
def update_message_bar(self, message):
    """Changes the message in the message bar item.

    :param message: Message to be updated
    :type message: str
    """
    message_bar_item = self.message_bar.createMessage(message)
    self.message_bar.pushWidget(message_bar_item, Qgs.Info)
```

update_priority_layers

Code:

```
update_priority_layers(update_groups=True)
```

Updates the priority weighting layers list in the UI.

Parameters:

Name	Type	Description
------	------	-------------

Default

update_groups bool Whether to update the priority groups list or not True

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

518
519
520  **Code:**
521
522
523
524
525
526 def update_priority_layers(self, update_groups=True):
527 """Updates the priority weighting layers list in the UI.
528
529 :param update_groups: Whether to update the priority groups list or not
530 :type update_groups: bool
531 """
532
533 self.priority_layers_list.clear()
534 for layer in settings_manager.get_priority_layers():
535 item = QtWidgets.QListWidgetItem()
536 item.setData(QtCore.Qt.DisplayRole, layer.get("name"))
537 item.setData(QtCore.Qt.UserRole, layer.get("uuid"))
538
539
540 self.priority_layers_list.addItem(item)
541 if update_groups:
542 for index in range(self.priority_groups_list.topLevelItemCount())
543 group = self.priority_groups_list.topLevelItem(index)
544 if group.text(0) in layer.get("groups"):
545 self.add_priority_layer_group(group, item)
546 else:
547 group_children = group.takeChildren()
548 children = []
549 for child in group_children:
550 if child.text(0) == layer.get("name"):
551 continue
552 children.append(child)
553 group.addChildren(children)
554
555
556 # Trigger check to enable/disable PWLs
557 self.on_extent_changed(self.extent_box.outputExtent())

`update_progress_bar` **Code:**`update_progress_bar(progress_dialog, value)`

Sets the value of the progress bar

Parameters:

Name	Type	Description	Default
progress_dialog	ProgressDialog	Dialog responsible for showing all the analysis operations progress.	required
value	float	Value to be set on the progress bar	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

```
1335
1336
1337  Code:
1338
1339
1340
1341
1342     def update_progress_bar(self, progress_dialog, value):
1343         """Sets the value of the progress bar
1344
1345         :param progress_dialog: Dialog responsible for showing
1346             all the analysis operations progress.
1347         :type progress_dialog: ProgressDialog
1348
1349         :param value: Value to be set on the progress bar
1350         :type value: float
1351         """
1352
1353         if progress_dialog and not self.processing_cancelled:
1354             try:
1355                 progress_dialog.update_progress_bar(int(value))
1356             except RuntimeError:
1357                 log(tr("Error setting value to a progress bar"), not
```

update_progress_dialog

 **Code:**

```
update_progress_dialog(progress_dialog, message=None)
```

Run report generation. This should be called after the analysis is complete.

Parameters:

Name	Type	Description	Default
progress_dialog	ProgressDialog	Dialog responsible for showing all the analysis operations progress.	required
message	ReportManager	Report manager used to generate analysis reports	None

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

```
1317
1318
1319  Code:
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333

def update_progress_dialog(
    self,
    progress_dialog,
    message=None,
):
    """Run report generation. This should be called after the
    analysis is complete.

    :param progress_dialog: Dialog responsible for showing
        all the analysis operations progress.
    :type progress_dialog: ProgressDialog

    :param message: Report manager used to generate analysis reports
    :type message: ReportManager
    """

    progress_dialog.change_status_message(message) if message is not None else None
```

update_pwl_layers**Code:**

```
update_pwl_layers(notify=False)
```

Updates the priority layers path available in the store implementation models

Parameters:

Name	Type	Description	Default
notify	bool	Whether to show message to user about the update	False

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py

262
263  **Code:**

264
265
266
267
268
269 def update_pwl_layers(self, notify=False):
270 """Updates the priority layers path available in
271 the store implementation models
272
273 :param notify: Whether to show message to user about the update
274 :type notify: bool
275 """
276 settings_manager.update_implementation_models()
277 self.update_priority_layers()
278 if notify:
279 self.show_message(
280 tr(
281 "Updated all the implementation models"
282 " with their respective priority layers"
283),
284 Qgis.Info,
285)
286 log(
287 tr(
288 "Updated all the implementation models"
289 " with their respective priority layers"
290)
291)

```
zoom_pilot_area
```

 **Code:**

```
zoom_pilot_area()
```

Zoom the current main map canvas to the pilot area extent.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/qgis_cplus_main.py`

 **Code:**

```
1372
1373
1374
1375
1376
1377
1378
1379 def zoom_pilot_area(self):
1380     """Zoom the current main map canvas to the pilot area extent."""
1381     map_canvas = iface.mapCanvas()
1382     extent_list = PILOT_AREA_EXTENT["coordinates"]
1383     default_extent = QgsRectangle(
1384         extent_list[0], extent_list[2], extent_list[1], extent_list[3]
1385     )
1386     zoom_extent = QgsRectangle(
1387         extent_list[0] - 0.5, extent_list[2], extent_list[1] + 0.5, ex
1388     )
1389
1390
1391     canvas_crs = map_canvas.mapSettings().destinationCrs()
1392     original_crs = QgsCoordinateReferenceSystem("EPSG:4326")
1393
1394
1395
1396     if canvas_crs.authid() != original_crs.authid():
1397         zoom_extent = self.transform_extent(zoom_extent, original_crs,
1398         default_extent = self.transform_extent(
1399             default_extent, original_crs, canvas_crs
1400         )
1401
1402
1403     aoi = QgsRubberBand(iface.mapCanvas(), QgsWkbTypes.PolygonGeometry)
1404
1405     aoi.setFillColor(QtGui.QColor(0, 0, 0, 0))
1406     aoi.setStrokeColor(QtGui.QColor(88, 128, 8))
1407     aoi.setWidth(3)
1408     aoi.setLineStyle(QtCore.Qt.DashLine)
1409
1410     geom = QgsGeometry.fromRect(default_extent)
1411
1412     aoi.setToGeometry(geom, canvas_crs)
```

```
map_canvas.setExtent(zoom_extent)  
map_canvas.refresh()
```

Carbon Layer Model

MVC model for carbon layer paths.

CarbonLayerItem

 **Code:**

```
CarbonLayerItem(layer_path)
```

Bases: **QStandardItem**

Represents a single carbon layer path.

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/carbon_item_model.py](#)

```
19  
20  
21  
22  
23  
24
```

```
def __init__(self, layer_path: str):  
    super().__init__()  
  
    self._layer_path = layer_path  
    self._is_valid = True  
    self.update(self._layer_path)
```

is_valid **property**

 **Code:**

```
is_valid
```

Returns the validity status of the carbon layer path.

The path could be invalid if it does not exist or if the corresponding map layer is invalid.

Returns:

Type Description

`bool` True if valid, else False.

`layer_path` `property`

 **Code:**

`layer_path`

Returns the path to the carbon layer.

Returns:

Type Description

`str` Path to the carbon layer.

`type`

 **Code:**

`type()`

Returns the type of the standard item.

Returns:

Type Description

`int` Type identifier of the carbob item.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/carbon_item_model.py

```
71
72  Code:
73
74
75
76
77
def type(self) -> int:
    """Returns the type of the standard item.

    :returns: Type identifier of the carbon item.
    :rtype: int
    """
    return QtGui.QStandardItem.UserType + 5
```

update

```
 Code:
update(layer_path)
```

Update the UI properties.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/carbon_item_model.py

47

48  **Code:**

49

50

51

52

53

54 def update(self, layer_path: str):

55 """Update the UI properties."""

56 self._layer_path = str(os.path.normpath(layer_path))

57 p = Path(self._layer_path)

58 self.setText(p.name)

59 self.setToolTip(self._layer_path)

60

61 # Check validity

62 if p.exists():

63 layer = QgsRasterLayer(layer_path)

64 if layer.isValid():

65 self._is_valid = True

66 self.setIcon(QtGui.QIcon())

67 else:

68 self._is_valid = False

69 error_icon = FileUtils.get_icon("mIndicatorLayerError")

70 self.setIcon(error_icon)

71 self.setToolTip(tr("Carbon layer is not invalid."))

72 else:

73 self._is_valid = False

74 error_icon = FileUtils.get_icon("mIndicatorLayerError.svg")

75 self.setIcon(error_icon)

76 self.setToolTip(tr("File path is invalid."))

CarbonLayerModel

 **Code:**

```
CarbonLayerModel(parent=None, carbon_paths=None)
```

Bases: `QStandardItemModel`

View model for carbon layers.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/carbon_item_model.py`

```
83
84  Code:
85
86
87
88
89
90
91
def __init__(
    self, parent=None, carbon_paths: typing.Union[typing.List[str], str] = None):
    super().__init__(parent)
    self.setColumnCount(1)

    if carbon_paths is not None:
        for cp in carbon_paths:
            self.add_carbon_layer(cp)
```

add_carbon_layer

 **Code:**

```
add_carbon_layer(layer_path)
```

Adds a carbon layer to the model.

Parameters:

Name	Type	Description	Default
layer_path	str	Carbon layer path.required	

Returns:

Type	Description
bool	True if the carbon layer was successfully added, else False if there is an existing item with the same path.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/carbon_item_model.py

93
94
95  **Code:**
96
97
98
99
100 def add_carbon_layer(self, layer_path: str) -> bool:
101 """Adds a carbon layer to the model.
102
103 :param layer_path: Carbon layer path.
104 :type layer_path: str
105
106 :returns: True if the carbon layer was successfully added,
107 else False if there is an existing item with the same path.
108 """
109
110 if self.contains_layer_path(layer_path):
111 return False
112
113 carbon_item = CarbonLayerItem(layer_path)
114 self.appendRow(carbon_item)
115
116 return True

carbon_layer_index

 **Code:**

```
carbon_layer_index(layer_path)
```

Get the model index for the given layer path.

Parameters:

Name	Type	Description	Default
layer_path	str	Carbon layer path.required	

Returns:

Type	Description
QtCore.QModelIndex	The index corresponding to the given layer path else an invalid index if not found.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/carbon_item_model.py

```
110  
111  
112      Code:  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134
```

```
def carbon_layer_index(self, layer_path: str) -> QtCore.QModelIndex:  
    """Get the model index for the given layer path.  
  
    :param layer_path: Carbon layer path.  
    :type layer_path: str  
  
    :returns: The index corresponding to the given layer path else  
    an invalid index if not found.  
    :rtype: QtCore.QModelIndex  
    """  
  
    norm_path = str(os.path.normpath(layer_path))  
    matching_index = None  
    for r in range(self.rowCount()):  
        index = self.index(r, 0)  
        if not index.isValid():  
            continue  
        item = self.itemFromIndex(index)  
        if item.layer_path == norm_path:  
            matching_index = index  
            break  
  
    if matching_index is None:  
        return QtCore.QModelIndex()  
  
    return matching_index
```

carbon_paths

 **Code:**

```
carbon_paths(valid_only=False)
```

Gets all the carbon paths in the model.

Parameters:

Name	Type	Description	Default
valid_only	bool	Only return the carbon paths that are valid.	False

Returns:

Type	Description
list	A collection of carbon paths in the model.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/carbon_item_model.py

```
151
152      Code:
153
154
155
156
157
158     def carbon_paths(self, valid_only: bool = False) -> list:
159         """Gets all the carbon paths in the model.
160
161         :param valid_only: Only return the carbon paths that are valid.
162         :type valid_only: bool
163
164         :returns: A collection of carbon paths in the model.
165         :rtype: list
166
167
168         """
169
170         carbon_paths = []
171         for r in range(self.rowCount()):
172             index = self.index(r, 0)
173             item = self.itemFromIndex(index)
174             if valid_only:
175                 if item.is_valid:
176                     carbon_paths.append(item.layer_path)
177             else:
178                 carbon_paths.append(item.layer_path)
179
180         return carbon_paths
```

contains_layer_path

 **Code:**

```
contains_layer_path(layer_path)
```

Checks if the specified layer path exists in the model.

Parameters:

Name	Type	Description	Default
layer_path	str	Carbon layer path.required	

Returns:

Type	Description
bool	True if the path exists, else False.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/carbon_item_model.py

```
136
137     🧑‍💻 Code:
138
139
140
141
142
143     def contains_layer_path(self, layer_path: str) -> bool:
144         """Checks if the specified layer path exists in the model.
145
146         :param layer_path: Carbon layer path.
147         :type layer_path: str
148
149
150             :returns: True if the path exists, else False.
151             :rtype: bool
152             """
153
154             carbon_idx = self.carbon_layer_index(layer_path)
155             if carbon_idx.isValid():
156                 return True
157
158
159             return False
```

update_carbon_path

 **Code:**

```
update_carbon_path(index, layer_path)
```

Update the carbon path at the given position specified by the index.

Parameters:

Name	Type	Description	Default
index	QModelIndex	Location to modify the carbon path.required	
layer_path	str	Carbon layer path.	required

Returns:

Type Description

bool	True if the path was successfully updated else False if there is no carbon item at the given location.
------	--

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/carbon_item_model.py

172
173  **Code:**

174
175
176
177
178
179 def update_carbon_path(self, index: QtCore.QModelIndex, layer_path: str) -> bool:
180 """Update the carbon path at the given position specified by the index.
181
182
183
184 :param index: Location to modify the carbon path.
185 :type index: QtCore.QModelIndex
186
187
188 :param layer_path: Carbon layer path.
189 :type layer_path: str
190
191
192
193 :returns: True if the path was successfully updated else False
194 if there is no carbon item at the given location.
195 :rtype: bool
196 """
197
198 if not index.isValid():
199 return False
200
201 item = self.itemFromIndex(index)
202 if item is None:
203 return False
204
205 item.update(layer_path)
206
207 return True

Component item model

Contains item models for view widgets such as NCS pathway or IM views.

ComponentItemModel

 **Code:**

```
ComponentItemModel(parent=None)
```

Bases: QStandardItemModel

View model for ModelComponent objects.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
641  
642  
643  
644  
645
```

```
 Code:  
  
def __init__(self, parent=None):  
    super().__init__(parent)  
    self.setColumnCount(1)  
  
    self._uuid_row_idx = {}
```

add_component_item

 **Code:**

```
add_component_item(component_item, position=-1)
```

Adds a model component item to the view model.

Parameters:

Name	Type	Description	Default
component_item	ModelComponentItem	Model component item to be added to the required view model.	
position	int	Reference row to insert the item.	-1

Returns:**Type Description**

bool	True if the component item was successfully added, else False if there is an existing component item with the same UUID.
------	--

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py`

Code:

```
def add_component_item(
    self, component_item: ModelComponentItem, position=-1
) -> bool:
    """Adds a model component item to the view model.

    :param component_item: Model component item to be added to the view
    model.
    :type component_item: ModelComponentItem

    :param position: Reference row to insert the item.
    :type position: int

    :returns: True if the component item was successfully added, else
    False if there is an existing component item with the same UUID.
    :rtype: bool
"""

    idx = position
    if position == -1:
        idx = self.rowCount()

    if self.contains_item(str(component_item.uuid)):
        return False

    self.insertRow(idx, component_item)

    self._re_index_rows()

    return True
```

component_item_by_uuid**Code:**

```
component_item_by_uuid(uuid_str)
```

Retrieves a ModelComponentItem based on a matching UUID.

Parameters:

Name	Type	Description	Default
uuid_str	str	UUID of the model item.required	

Returns:

Type	Description
ModelComponentItem	Component item matching the given UUID or None if not found.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703

 **Code:**

```
def component_item_by_uuid(
    self, uuid_str: str
) -> typing.Union[ModelComponentItemType, None]:
    """Retrieves a ModelComponentItem based on a matching UUID.

    :param uuid_str: UUID of the model item.
    :type uuid_str: str

    :returns: Component item matching the given UUID or None if not found.
    :rtype: ModelComponentItem
    """
    if uuid_str not in self._uuid_row_idx:
        return None

    row = self._uuid_row_idx[uuid_str]

    return self.item(row)
```

contains_item

 **Code:**

```
contains_item(item_uuid)
```

Checks if the model contains an item with the given UUID.

Parameters:

Name	Type	Description	Default
item_uuid	str	UUID of the model item.required	

Returns:

Type	Description
bool	True if there is an existing item else False.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

676

677

678

679

680

681

682

683

684

685

 **Code:**

```
def contains_item(self, item_uuid: str) -> bool:  
    """Checks if the model contains an item with the given UUID.  
  
    :param item_uuid: UUID of the model item.  
    :type item_uuid: str  
  
    :returns: True if there is an existing item else False.  
    :rtype: bool  
    """  
  
    return True if self.component_item_by_uuid(item_uuid) is not None else False
```

enable_default_items

 **Code:**

```
enable_default_items(state)
```

Enable or disable items for default model components.

Parameters:

Name	Type	Description	Default
state	bool	True to enable or False to disable.required	

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
800
801  Code:
802
803
804
805
806
807     def enable_default_items(self, state: bool):
808         """Enable or disable items for default model components.
809
810         :param state: True to enable or False to disable.
811         :type state: bool
812         """
813
814         for item in self.model_component_items():
815             if not isinstance(item, LayerComponentItem):
816                 continue
817
818             if not item.user_defined:
819                 item.setEnabled(state)
```

index_by_uuid

 **Code:**

```
index_by_uuid(uuid_str)
```

Get the QModelIndex object for the component item matching the given UUID identifier.

Parameters:

Name	Type	Description	Default
uuid_str	str	UUID of the model item.required	

Returns:

Type	Description
QtCore.QModelIndex	QModelIndex for the component item matching the given UUID or an invalid QModelIndex if not found.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721

 **Code:**

```
def index_by_uuid(self, uuid_str) -> QtCore.QModelIndex:  
    """Get the QModelIndex object for the component item matching the  
    given UUID identifier.  
  
    :param uuid_str: UUID of the model item.  
    :type uuid_str: str  
  
    :returns: QModelIndex for the component item matching the given  
    UUID or an invalid QModelIndex if not found.  
    :rtype: QtCore.QModelIndex  
    """  
  
    if uuid_str not in self._uuid_row_idx:  
        return QtCore.QModelIndex()  
  
    row = self._uuid_row_idx[uuid_str]  
  
    return self.index(row, 0)
```

model_component_items

 **Code:**

```
model_component_items()
```

Returns model component items in the model.

Returns:**Type Description**

`list` Model component items in the model.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py`

748

749

750

751

752

753

754

755

756

757

758

759

760

761

Code:

```
def model_component_items(self) -> typing.List[ModelComponentItem]:
    """Returns model component items in the model.

    :returns: Model component items in the model.
    :rtype: list
    """

    rows = self.rowCount()

    items = []
    for r in range(rows):
        item = self.item(r)
        items.append(item)

    return items
```

model_components

Code:

```
model_components()
```

Returns a collection of model component objects in the model.

Returns:**Type Description**

list A collection of all model component objects.

- Source code in /home/samwel1/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

740
741
742
743
744
745
746

Code:

```
def model_components(self) -> typing.List[BaseModelComponentType]:  
    """Returns a collection of model component objects in the model.  
  
    :returns: A collection of all model component objects.  
    :rtype: list  
    """  
  
    return [item.model_component for item in self.model_component_items]
```

remove_component_item**Code:**

```
remove_component_item(uuid_str)
```

Removes a ModelComponentItem based on a matching UUID.

Parameters:

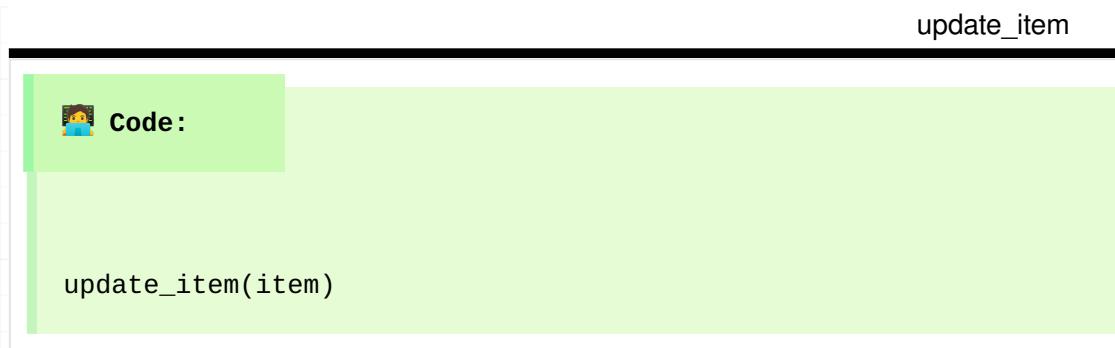
Name	Type	Description	Default
uuid_str	str	UUID of the model item to be removed.	required

Returns:**Type Description**

bool True if the component item was successfully removed, else False if there was not matching UUID.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

777
778  **Code:**
779
780
781
782
783
784 def remove_component_item(self, uuid_str: str) -> bool:
785 """Removes a ModelComponentItem based on a matching UUID.
786
787 :param uuid_str: UUID of the model item to be removed.
788 :type uuid_str: str
789
790 :returns: True if the component item was successfully removed, else
791 False if there was not matching UUID.
792 :rtype: bool
793 """
794
795 if not self.contains_item(uuid_str):
796 return False
797
798 if uuid_str not in self._uuid_row_idx:
799 return False
800
801 self.removeRows(self._uuid_row_idx[uuid_str], 1)
802 del self._uuid_row_idx[uuid_str]
803
804 self._re_index_rows()
805
806 return True



Update an existing ModelComponentItem if it exists in the model.

Parameters:

Name	Type	Description	Default
item	ModelComponentItemType	An updated instance of the ModelComponentItem.required	

Returns:

Type	Description
bool	True if the item was successfully updated, else False if there was no matching item found in the model.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
723
724      Code:
725
726
727
728
729
730     def update_item(self, item: ModelComponentItemType) -> bool:
731         """Update an existing ModelComponentItem if it exists in the model.
732
733         :param item: An updated instance of the ModelComponentItem.
734         :type item: ModelComponentItem
735
736         :returns: True if the item was successfully updated, else False
737         if there was no matching item found in the model.
738         :rtype: bool
739
740         """
741
742         if not self.contains_item(item.uuid):
743             return False
744
745             item.update(item.model_component)
746
747             return True
```

IMItemModel

Bases: [ComponentItemModel](#)

View model for implementation model.

[add_implementation_model](#)

```
     Code:
```

```
    add_implementation_model(implementation_model, layer=None)
```

Add an ImplementationModel object to the model.

Parameters:

Name	Type	Description	Default
implementation_model	ImplementationModel	ImplementationModel object to be added to the view.	required
layer	QgsMapLayer	Map layer for the implementation model.	None

Returns:**Type Description**

bool	True if ImplementationModel object was added successfully, else False.
------	--

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017 **Code:**

```
def add_implementation_model(
    self, implementation_model: ImplementationModel, layer: QgsMapLayer = None
) -> bool:
    """Add an ImplementationModel object to the model.

    :param implementation_model: ImplementationModel object to be
        added to the view.
    :type implementation_model: ImplementationModel

    :param layer: Map layer for the implementation model.
    :type layer: QgsMapLayer

    :returns: True if ImplementationModel object was added
        successfully, else False.
    :rtype: bool
    """

    # Check if we can retrieve the layer from the path
    if layer is None:
        if implementation_model.path:
            layer = implementation_model.to_map_layer()

    implementation_model_item = ImplementationModelItem.create(implementation_
result = self.add_component_item(implementation_model_item)
if layer:
    status = self.set_model_layer(implementation_model_item, layer)
    if not status:
        result = False
else:
    # Add NCS pathways. If there are underlying NCS pathway objects then
    # clone them, remove then re-insert so that the underlying NCS pathways
    # have the unique UUID in the IM item.
```

```
if result:
    cloned_implementation_model = clone_implementation_model(
        implementation_model
    )
    cloned_ncs_pathways = cloned_implementation_model.pathways

    # Remove pathways in the IM
    implementation_model.pathways = []

    # Now add the NCSs afresh
    for ncs in cloned_ncs_pathways:
        ncs_item = NcsPathwayItem.create(ncs)
        self.add_ncs_pathway(ncs_item, implementation_model_item)

return result
```

add_ncs_pathway

Code:

```
add_ncs_pathway(ncs_item, target_model_item)
```

Adds an NCS pathway item to the model.

Parameters:

Name	Type	Description	Default
ncs_item	NcsPathwayItem	NCS pathway item to the collection.	required
target_model_item	ImplementationModelItem	Target implementation model for the NCS item.	required

Returns:

Type Description

bool	True if the NCS pathway item was successfully added, else False if the underlying NCS pathway object was invalid, there is an existing item with the same UUID or if there is already a map layer defined for the implementation model.
------	---

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

1091
1092
1093  **Code:**
1094
1095
1096
1097
1098 def add_ncs_pathway(
1099 self, ncs_item: NcsPathwayItem, target_model_item: ImplementationModelItem
1100) -> bool:
1101 """Adds an NCS pathway item to the model.
1102
1103 :param ncs_item: NCS pathway item to the collection.
1104 :type ncs_item: NcsPathwayItem
1105
1106 :param target_model_item: Target implementation model for the NCS item.
1107 :type target_model_item: ImplementationModelItem
1108
1109 :returns: True if the NCS pathway item was successfully added, else
1110 False if there underlying NCS pathway object was invalid, there
1111 is an existing item with the same UUID or if there is already
1112 a map layer defined for the implementation model.
1113 """
1114
1115 idx = target_model_item.index()
1116 if not idx.isValid():
1117 return False
1118
1119 if not isinstance(target_model_item, LayerComponentItem):
1120 return False
1121
1122 # Do not add if the IM item has been disabled (e.g. disabled default IMs)
1123 if not target_model_item.isEnabled():
1124 return False
1125
1126 # If there is an existing layer then return
1127 if target_model_item.layer:
1128 return False

```
clone_ncs_item = ncs_item.clone()
status = target_model_item.add_ncs_pathway_item(clone_ncs_item)
if not status:
    return False

bottom_idx = target_model_item.bottom_ncs_item_index()
reference_row = max(bottom_idx.row(), idx.row())
self.add_component_item(clone_ncs_item, reference_row + 1)

self.im_pathways_updated.emit(target_model_item)

return True
```

dropMimeData

Code:

```
dropMimeData(data, action, row, column, parent)
```

Implements behaviour for handling data supplied by drag and drop operation.

Parameters:

Name	Type	Description	Default
data	QMimeType	Object containing data from the drag operation.	required
action	DropAction	Type of the drag and drop operation.	required
row	int	Row location of dropped data.	required
column	int	Column location of dropped data.	required
parent	QModelIndex	Index location for target item where the operation ended.	required

Returns:

Type Description

bool True if the data and action can be handled by the model, else False.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

1290
1291  **Code:**
1292
1293
1294
1295
1296
1297 **def dropMimeData(**
1298 **self,**
1299 **data: QtCore.QMimeData,**
1300 **action: QtCore.Qt.DropAction,**
1301 **row: int,**
1302 **column: int,**
1303 **parent: QtCore.QModelIndex,**
1304 **) -> bool:**
1305 `"""Implements behaviour for handling data supplied by drag and drop operation.`
1306
1307 **:param data:** Object containing data from the drag operation.
1308 **:type data:** QtCore.QMimeData
1309
1310 **:param action:** Type of the drag and drop operation.
1311 **:type action:** QtCore.Qt.DropAction
1312
1313 **:param row:** Row location of dropped data.
1314 **:type row:** int
1315
1316 **:param column:** Column location of dropped data.
1317 **:type column:** int
1318
1319 **:param parent:** Index location for target item where the
1320 **operation ended.**
1321 **:type parent:** QtCore.QModelIndex
1322
1323 **:returns:** True if the data and action can be handled by the
1324 **model, else False.**
1325 **:rtype:** bool
1326 `"""`
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346

```
1347     if action == QtCore.Qt.IgnoreAction:
1348         return True
1349
1350
1351     if not data.hasFormat(NCS_MIME_TYPE):
1352         return False
1353
1354
1355     encoded_data = data.data(NCS_MIME_TYPE)
1356     data_stream = QtCore.QDataStream(encoded_data, QtCore.QIODevice.ReadOnly)
1357
1358     ncs_items = []
1359
1360
1361     while not data_stream.atEnd():
1362         byte_data = QtCore.QByteArray()
1363         data_stream >> byte_data
1364
1365
1366         item_data = json.loads(byte_data.data())
1367         ncs_pathway = create_ncs_pathway(item_data)
1368
1369
1370         ncs_item = NcsPathwayItem(ncs_pathway)
1371         ncs_items.append(ncs_item)
1372
1373
1374         # Get reference ImplementationModel item
1375         if parent.isValid():
1376             model_item = self.itemFromIndex(parent)
1377         else:
1378             row_count = self.rowCount()
1379             model_item = self.item(row_count - 1)
1380
1381
1382         if model_item is None or isinstance(model_item, LayerItem):
1383             return False
1384
1385
1386         if model_item.type() == NCS_PATHWAY_TYPE:
1387             target_im_item = model_item.parent
1388         else:
1389             target_im_item = model_item
```

```
# Add NCS items to model.  
status = True  
for item in ncs_items:  
    status = self.add_ncs_pathway(item, target_im_item)  
  
return status
```

model_items

 **Code:**

model_items()

Returns all ImplementationModelItem objects in the model.

Returns:

Type Description

- `list` Implementation model items in the model.
- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py`

1201
1202
1203
1204
1205
1206
1207
1208
1209

 **Code:**

```
def model_items(self) -> typing.List[ImplementationModelItem]:  
    """Returns all ImplementationModelItem objects in the model.  
  
    :returns: Implementation model items in the model.  
    :rtype: list  
    """  
  
    component_items = self.model_component_items()  
  
    return [ci for ci in component_items if ci.type() == IMPLEMENT
```

models**Code:**`models()`

Returns implementation model objects in the model.

Returns:**Type Description**

`list` Implementation model objects in the model.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py`

1193
1194
1195
1196
1197
1198
1199

```
def models(self) -> typing.List[ImplementationModel]:  
    """Returns implementation model objects in the model.  
  
    :returns: Implementation model objects in the model.  
    :rtype: list  
    """  
  
    return [model_item.implementation_model for model_item in se
```

remove_implementation_model**Code:**`remove_implementation_model(uuid_str)`

Remove an implementation model item from the model.

param uuid: UUID of the implementation model item to be removed.

Returns:

Type Description

bool True if the implementation model item was successfully removed, else False if there was not matching UUID.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py`

1260

1261

1262

1263

1264

1265

1266

1267

1268

1269

1270

1271

1272

1273

1274

1275

1276

1277

1278

1279

1280

1281

1282

1283

 **Code:**

```
def remove_implementation_model(self, uuid_str: str) -> bool:
    """Remove an implementation model item from the model.

    param uuid: UUID of the implementation model item to
    be removed.
    :type uuid_str: str

    :returns: True if the implementation model item was successfully
    removed, else False if there was not matching UUID.
    :rtype: bool
    """

    implementation_model_item = self.component_item_by_uuid(uuid_str)
    if implementation_model_item is None:
        return False

    if len(implementation_model_item.ncs_items) > 0:
        ncs_items = implementation_model_item.ncs_items
        for item in ncs_items:
            self.remove_component_item(item.uuid)
    else:
        # Layer item
        self.remove_layer(implementation_model_item)

    return self.remove_component_item(uuid_str)
```

remove_layer

 **Code:**

```
remove_layer(implementation_model_item)
```

Removes the layer reference from the underlying implementation model.

Parameters:

Name	Type	Description	Default
implementation_model_item	ImplementationModelItem	Implementation model item whose layer is to be removed.	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040
```

 **Code:**

```
def remove_layer(self, implementation_model_item: Implementation
    """Removes the layer reference from the underlying
    implementation model.

    :param implementation_model_item: Implementation model
    item whose layer is to be removed.
    :type implementation_model_item: ImplementationModelItem
    """
    if implementation_model_item.layer is None:
        return

    if not self.contains_item(implementation_model_item.uuid):
        return

    # Remove item in model
    item_idx = self.index_by_uuid(implementation_model_item.uuid)
    layer_row = item_idx.row() + 1
    self.removeRows(layer_row, 1)
    self._re_index_rows()

    # Remove underlying layer reference
    implementation_model_item.clear_layer()
```

remove_ncs_pathway_item

 **Code:**

```
remove_ncs_pathway_item(ncs_uuid, parent)
```

Remove an NCS pathway item from the model.

param uuid: UUID of the NCS pathway item to be removed.

Parameters:

Name	Type	Description	Default
parent	ImplementationModelItem	Reference implementation model item that is the parent to the NCS pathway item.	required

Returns:

Type Description

bool	True if the NCS pathway item has been successfully removed, else False if there was no matching UUID.
------	---

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158
```

 **Code:**

```
def remove_ncs_pathway_item(  
    self, ncs_uuid: str, parent: ImplementationModelItem  
) -> bool:  
    """Remove an NCS pathway item from the model.  
  
    param uuid: UUID of the NCS pathway item to be removed.  
    :type ncs_uuid: str  
  
    :param parent: Reference implementation model item that  
    is the parent to the NCS pathway item.  
    :type parent: ImplementationModelItem  
  
    :returns: True if the NCS pathway item has been  
    successfully removed, else False if there was  
    no matching UUID.  
    :rtype: bool  
    """  
  
    status = parent.remove_ncs_pathway_item(ncs_uuid)  
    if not status:  
        return False  
  
    self.im_pathways_updated.emit(parent)  
  
    return self.remove_component_item(ncs_uuid)
```

remove_ncs_pathway_items

 **Code:**

```
remove_ncs_pathway_items(ncs_pathway_uuid)
```

Delete NCS pathway items matching the given NCS pathway model.

If the NCS pathway model is not valid then the NCS pathway items in the implementation model item will not be deleted.

Parameters:

Name	Type	Description	Default
ncs_pathway_uuid	str	Unique identifier of the NCS pathway object whose corresponding models are to be removed in the implementation models.	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

1240
1241
1242  **Code:**
1243
1244
1245
1246
1247
1248 def remove_ncs_pathway_items(self, ncs_pathway_uuid: str):
1249 """Delete NCS pathway items matching the given NCS pathway model.
1250
1251 If the NCS pathway model is not valid then the NCS pathway items
1252 in the implementation model item will not be deleted.
1253
1254
1255 :param ncs_pathway_uuid: Unique identifier of the NCS pathway object
1256 whose corresponding models are to be removed in the implementation
1257 models.
1258 :type ncs_pathway_uuid: str
1259 """
1260
1261 for im_item in self.model_items():
1262 ncs_item_for_original = im_item.ncs_item_from_original_pathway(
1263 ncs_pathway_uuid
1264)
1265 if ncs_item_for_original is None:
1266 continue
1267
1268 status = self.remove_ncs_pathway_item(ncs_item_for_original.uuid,
1269

set_model_layer

Code:

```
set_model_layer(  
    implementation_model_item, layer, display_name=""  
)
```

Set the layer for the given implementation model item.

Parameters:

Name	Type	Description	Default
implementation_model_item	ImplementationModelItem	Implementation model item whose layer is to be specified.	required
layer	QgsMapLayer	Map layer to be set required for the implementation model.	
display_name	str	Display name for the layer node. If not specified then the name from the map layer is used.	' '

Returns:

Type Description

bool	True if the layer was successfully set for the implementation model, else False if the layer is invalid, if there are already existing NCS pathways in the implementation model or if the item is not in the model.
------	---

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

1042
1043
1044  **Code:**
1045
1046
1047
1048
1049 def set_model_layer(
1050 self,
1051 implementation_model_item: ImplementationModelItem,
1052 layer: QgsMapLayer,
1053 display_name: str = "",
1054) -> bool:
1055 """Set the layer for the given implementation model item.
1056
1057 :param implementation_model_item: Implementation model item
1058 whose layer is to be specified.
1059 :type implementation_model_item: ImplementationModelItem
1060
1061 :param layer: Map layer to be set for the implementation model.
1062 :type layer: QgsMapLayer
1063
1064 :param display_name: Display name for the layer node. If not
1065 specified then the name from the map layer is used.
1066 :type display_name: str
1067
1068 :returns: True if the layer was successfully set for the
1069 implementation model, else False if the layer is invalid, if
1070 there are already existing NCS pathways in the implementation
1071 model or if the item is not in the model.
1072 :rtype: bool
1073 """
1074 if len(implementation_model_item.ncs_items) > 0:
1075 return False
1076
1077 if not self.contains_item(implementation_model_item.uuid):
1078 return False

```
if not implementation_model_item.set_layer(layer):
    return False

if not display_name:
    display_name = layer.name()

icon = FileUtils.get_icon("mIconRaster.svg")
item = LayerItem(icon, display_name)
item.setToolTip(layer.source())
item.setData(implementation_model_item)

item_idx = self.index_by_uuid(implementation_model_item.uuid)
layer_row = item_idx.row() + 1
self.insertRow(layer_row, item)
self._re_index_rows()

return True
```

update_implementation_model

Code:

```
update_implementation_model(
    implementation_model, layer=None
)
```

Updates the implementation model item in the model.

Parameters:

Name	Type	Description	Default
implementation_model	ImplementationModel	implementation_model object whose corresponding item is to be updated.	required
layer	QgsMapLayer	Map layer to be updated for the implementation if specified.	None

Returns:**Type Description**

bool Returns True if the operation was successful else False if the matching item was not found in the model.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

1160
1161  **Code:**
1162
1163
1164
1165
1166
1167 def update_implementation_model(
1168 self, implementation_model: ImplementationModel, layer: QgsMapLayer
1169) -> bool:
1170 """Updates the implementation model item in the model.
1171
1172 :param implementation_model: implementation_model object whose
1173 corresponding item is to be updated.
1174 :type implementation_model: ImplementationModel
1175
1176 :param layer: Map layer to be updated for the
1177 implementation if specified.
1178 :type layer: QgsMapLayer
1179
1180
1181 :returns: Returns True if the operation was successful else False
1182 if the matching item was not found in the model.
1183 """
1184
1185 item = self.component_item_by_uuid(str(implementation_model.uuid))
1186 if item is None:
1187 return False
1188
1189 status = self.update_item(item)
1190 if not status:
1191 return False
1192
1193 # Update layer information
1194 self.remove_layer(item)
1195 if layer:
1196 layer_status = self.set_model_layer(item, layer)
1197 if not layer_status:
1198 return False

```
return True
```

update_ncs_pathway_items

Code:

```
update_ncs_pathway_items(ncs_pathway)
```

Update NCS pathway items matching the given NCS pathway model.

If the NCS pathway model is not valid then the NCS pathway items in the implementation model item will not be updated.

Parameters:

Name	Type	Description	Default
ncs_pathway	NcsPathway	Original NCS pathway object whose corresponding models required are to be updated.	

Returns:

Type Description

bool True if matching NCS pathway items have been updated, else False.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

1211
1212  **Code:**

1213
1214
1215
1216
1217
1218 def update_ncs_pathway_items(self, ncs_pathway: NcsPathway) -> bool:
1219 """Update NCS pathway items matching the given NCS pathway model.
1220
1221 If the NCS pathway model is not valid then the NCS pathway items
1222 in the implementation model item will not be updated.
1223
1224
1225 :param ncs_pathway: Original NCS pathway object whose corresponding
1226 models are to be updated.
1227 :type ncs_pathway: NcsPathway
1228
1229 :returns: True if matching NCS pathway items have been updated,
1230 else False.
1231 :rtype: bool
1232 """
1233
1234 if not ncs_pathway.is_valid():
1235 return False
1236
1237
1238 for im_item in self.model_items():
1239 ncs_item_for_original = im_item.ncs_item_from_original_pathway(ncs_pathway)
1240 if ncs_item_for_original is None:
1241 continue
1242
1243 item_pathway = ncs_item_for_original.ncs_pathway
1244 # Copy attribute values excluding the UUID
1245 copy_layer_component_attributes(item_pathway, ncs_pathway)
1246 ncs_item_for_original.update(item_pathway)
1247
1248 return True

ImplementationModelItem

 **Code:**

```
ImplementationModelItem(implementation_model)
```

Bases: [LayerComponentItem](#)

Standard item for an implementation model object.

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py](#)

310
311
312
313
314
315
316
317
318
319
320
321
322

```
def __init__(self, implementation_model: ImplementationModel):
    super().__init__(implementation_model)
    self._implementation_model = implementation_model

    font = self.font()
    font.setBold(True)
    self.setFont(font)

    self._ncs_items = []

    # Remap pathway uuids so that there are no duplicate
    # pathways under each implementation model.
    self._uuid_remap = {}
```

implementation_model [property] **Code:**`implementation_model`

Returns an instance of the underlying ImplementationModel object.

Returns:

Type	Description
------	-------------

`ImplementationModel` The underlying ImplementationModel object.

layer_item [property] **Code:**`layer_item`

Returns the view item for the layer.

Returns:

Type	Description
------	-------------

`QtGui.QStandardItem` Returns the view item for the map layer else None if no layer has been specified for the model.

ncs_items [property] **Code:**`ncs_items`

Returns a collection of NcsPathwayItem in this implementation model.

Returns:**Type Description**

`list` Collection of NcsPathwayItem objects in this implementation model.

`ncs_pathways` `property`



`ncs_pathways`

Returns a collection of NcsPathway objects.

Returns:**Type Description**

`list` Collection of NcsPathway objects linked to the underlying ImplementationModel object.

`original_ncs_pathways` `property`



`original_ncs_pathways`

Returns a collection of NcsPathway objects but with their original UUIDs.

These are used for persisting the NCsPathway objects related to the underlying IM object.

Returns:**Type Description**

`list` Collection of NcsPathway objects with their original UUIDs linked to the underlying ImplementationModel object.

add_ncs_pathway_item

 **Code:**

```
add_ncs_pathway_item(ncs_item)
```

Adds an NCS pathway item to this implementation model item.

If the item already contains a layer, then the add operation will not be successful.

Parameters:

Name	Type	Description	Default
ncs_item	NcsPathwayItem	NCS pathway item to the collection.required	

Returns:

Type	Description
bool	True if the NCS pathway item was successfully added, else False if the underlying NCS pathway object was invalid, there is an existing item with the same UUID or if the layer property had already been set.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

460
461  **Code:**

462
463
464
465
466
467 **def add_ncs_pathway_item(self, ncs_item: NcsPathwayItem) -> bool:**
468 <="" data-bbox="238 268 800 295"/>
469 <="" data-bbox="238 325 848 352"/>
470 <="" data-bbox="238 375 765 388"/>
471 <="" data-bbox="238 400 545 413"/>
472
473
474 <="" data-bbox="238 325 848 352"/>
475 <="" data-bbox="238 375 765 388"/>
476
477
478 <="" data-bbox="238 325 848 352"/>
479 <="" data-bbox="238 375 765 388"/>
480 <="" data-bbox="238 400 545 413"/>
481
482
483 <="" data-bbox="238 400 920 505"/>
484
485
486
487
488
489
490
491 <="" data-bbox="238 505 380 518"/>
492 <="" data-bbox="238 530 405 543"/>
493
494
495
496 <="" data-bbox="238 585 485 598"/>
497 <="" data-bbox="238 600 425 613"/>
498 <="" data-bbox="238 615 605 628"/>
499
500
501
502 <="" data-bbox="238 660 560 673"/>
 <="" data-bbox="238 685 405 698"/>

 <="" data-bbox="238 720 655 733"/>
 <="" data-bbox="238 735 405 748"/>

 <="" data-bbox="238 770 515 783"/>
 <="" data-bbox="238 785 405 798"/>

```
if self._implementation_model.contains_pathway(ncs_item.uuid):
    return False

if not self._implementation_model.add_ncs_pathway(ncs_item.ncs_pathwa
    return False

self._ncs_items.append(ncs_item)
ncs_item._parent = self

self._uuid_remap[old_uuid] = str(new_uuid)

return True
```

bottom_ncs_item_index

 **Code:**

bottom_ncs_item_index()

Returns the model index of the bottom-most NcsPathwayItem under this implementation model item.

Returns:

Type **Description**

QModelIndex Model index of the bottom-most NcsPathwayItem.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545
```

 **Code:**

```
def bottom_ncs_item_index(self) -> typing.Union[QtCore.QModelIndex,  
                                                 None]:  
    """Returns the model index of the bottom-most NcsPathwayItem  
    under this implementation model item.  
  
    :returns: Model index of the bottom-most NcsPathwayItem.  
    :rtype: QModelIndex  
    """  
  
    if len(self._ncs_items) == 0:  
        return None  
  
    bottom_ncs_item = max(self._ncs_items, key=lambda n: n.index())  
  
    return bottom_ncs_item.index()
```

clear_layer

 **Code:**

```
clear_layer()
```

Clears the layer reference in the model component.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

423
424
425

 **Code:**

```
def clear_layer(self):
    """Clears the layer reference in the model component."""
    self._implementation_model.clear_layer()
```

clone

 **Code:**

```
clone()
```

Creates a cloned version of this item.

The cloned IM will contain pathways with the original UUID. The UUID of the IM will not change.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

566
567
568
569
570
571
572
573
574
575
576
577
578

 **Code:**

```
def clone(self) -> "ImplementationModelItem":  
    """Creates a cloned version of this item.  
  
    The cloned IM will contain pathways with the  
    original UUID. The UUID of the IM will not change.  
    """  
  
    implementation_model = clone_implementation_model(  
        self.implementation_model,  
    )  
    # Use NCS pathways with original UUIDs  
    implementation_model.pathways = self.original_ncs_pathways  
  
    return ImplementationModelItem(implementation_model)
```

contains_ncs_item

 **Code:**

```
contains_ncs_item(item_uuid)
```

Checks whether this item contains an NcsPathway item with the given UUID.

Parameters:

Name	Type	Description	Default
item_uuid	str	UUID of the NcsPathway item to search for.	required

Returns:

Type Description

bool True if there is an NcsPathwayItem matching the given UUID, else False.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
444
445  Code:
446
447
448
449
450
451     def contains_ncs_item(self, item_uuid: str) -> bool:
452         """Checks whether this item contains an NcsPathway item with
453         the given UUID.
454
455         :param item_uuid: UUID of the NcsPathway item to search for.
456         :type item_uuid: str
457
458
459             :returns: True if there is an NcsPathwayItem matching the
460             given UUID, else False.
461             :rtype: bool
462
463
464             if self.ncs_item_by_uuid(item_uuid) is None:
465                 return False
466
467
468             return True
```

create staticmethod

 **Code:**

```
create(implementation_model)
```

Creates an instance of the ImplementationModelItem from the model object.

Returns:

Type

Description

ImplementationModel	An instance of the ImplementationModelItem item to be used in a standard model.
----------------------------	---

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py`

```
555
556  Code:
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
```

enable default pathways

Code :

enable default pathways(state)

Enable or disable default NCS pathway items.

Parameters:

Name	Type	Description	Default
state	bool	True to enable default NCS pathways else False to disable them.required	

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
580
581      Code:
582
583
584
585
586
587     def enable_default_pathways(self, state: bool):
588         """Enable or disable default NCS pathway items.
589
590         :param state: True to enable default NCS pathways else False
591         to disable them.
592         :type state: bool
593         """
594
595         for ncs_item in self._ncs_items:
596             if ncs_item.user_defined:
597                 continue
598
599             if ncs_item.isEnabled() != state:
600                 ncs_item.setEnabled(state)
```

ncs_item_by_uuid

 **Code:**

```
ncs_item_by_uuid(ncs_uuid)
```

Returns an NcsPathway item matching the given UUID.

Parameters:

Name	Type	Description	Default
ncs_uuid	str	UUID of the NcsPathway item to retrieve.required	

Returns:

Type	Description
NcsPathwayItem	NcsPathwayItem matching the given UUID, else None if not found.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
427
428     Code:
429
430
431
432
433
434     def ncs_item_by_uuid(self, ncs_uuid: str) -> typing.Union[NcsPathwayItem, None]:
435         """Returns an NcsPathway item matching the given UUID.
436
437         :param ncs_uuid: UUID of the NcsPathway item to retrieve.
438         :type ncs_uuid: str
439
440         :returns: NcsPathwayItem matching the given UUID, else None
441         if not found.
442         :rtype: NcsPathwayItem
443
444         """
445
446         ncs_items = [n for n in self._ncs_items if n.uuid == ncs_uuid]
447
448         if len(ncs_items) == 0:
449             return None
450
451         return ncs_items[0]
```

ncs_item_from_original_pathway

Code:

```
ncs_item_from_original_pathway(ncs_pathway)
```

Retrieves the NCS item corresponding to the original NCS pathway i.e. before it is added to this implementation model item.

Parameters:

Name	Type	Description	Default
ncs_pathway	Union[NcsPathway, str]	Original NCS pathway data model or unique identifier of the NCS pathway.	required

Returns:

Type	Description
Union[NcsPathwayItem, None]	The matching NCS pathway item in this implementation model item, else None if there is no matching item.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378
```

 **Code:**

```
def ncs_item_from_original_pathway(  
    self, ncs_pathway: typing.Union[NcsPathway, str]  
) -> typing.Union[NcsPathwayItem, None]:  
    """Retrieves the NCS item corresponding to the original NCS  
    pathway i.e. before it is added to this implementation  
    model item.  
  
    :param ncs_pathway: Original NCS pathway data model or  
    unique identifier of the NCS pathway.  
    :type ncs_pathway: NcsPathway, str  
  
    :returns: The matching NCS pathway item in this implementation  
    model item, else None if there is no matching item.  
    """  
  
    if isinstance(ncs_pathway, NcsPathway):  
        ncs_uuid = str(ncs_pathway.uuid)  
    else:  
        ncs_uuid = ncs_pathway  
  
    if ncs_uuid not in self._uuid_remap:  
        return None  
  
    new_uuid = self._uuid_remap[ncs_uuid]  
  
    return self.ncs_item_by_uuid(new_uuid)
```

remove_ncs_pathway_item

 **Code:**

```
remove_ncs_pathway_item(item_uuid)
```

Removes the NcsPathwayItem matching the given UUID.

Parameters:

Name	Type	Description	Default
item_uuid	str	The UUID of the NcsPathwayItem to remove.	required

Returns:

Type	Description
bool	True if the item was successfully removed, else False.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

504
505
506  **Code:**
507
508
509
510
511 def remove_ncs_pathway_item(self, item_uuid: str) -> bool:
512 """Removes the NcsPathwayItem matching the given UUID.
513
514
515
516 :param item_uuid: The UUID of the NcsPathwayItem to remove.
517 :type item_uuid: str
518
519
520 :returns: True if the item was successfully removed, else
521 False.
522 :rtype: bool
523 """
524 if not self.contains_ncs_item(item_uuid):
525 return False
526
527 idxs = [i for i, n in enumerate(self._ncs_items) if n.uuid ==
528 item_uuid]
529 if len(idxs) == 0:
530 return False
531
532 item = self._ncs_items.pop(idxs[0])
533 item._parent = None
534 del item
535
536 self._implementation_model.remove_ncs_pathway(item_uuid)
537
538 old_uuids = [k for k, v in self._uuid_remap.items() if v == item_uuid]
539 if len(old_uuids) > 0:
540 del self._uuid_remap[old_uuids[0]]
541
542 return True

setEnabled **Code:**`setEnabled(enabled)`

Override for default implementation that also enables or disables NCS pathway items.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py`

```
614  
615  
616  
617  
618  
619  
620  
621
```

 **Code:**

```
def setEnabled(self, enabled: bool):  
    """Override for default implementation that  
    also enables or disables NCS pathway items.  
    """  
    self.enable_default_pathways(enabled)  
    self._enable_layer_item(enabled)  
    self.setSelectable(enabled)  
    super().setEnabled(enabled)
```

type **Code:**`type()`

Returns the type of the standard item.

Returns:**Type Description**

`int` Type identifier of the standard item.

- Source code in `/home/samwel1/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py`

547
548
549
550
551
552
553

 **Code:**

```
def type(self) -> int:  
    """Returns the type of the standard item.  
  
    :returns: Type identifier of the standard item.  
    :rtype: int  
    """  
  
    return IMPLEMENTATION_MODEL_TYPE
```

LayerComponentItem **Code:**

```
LayerComponentItem(model_component)
```

Bases: `ModelComponentItem`

Base class view item for layer-based component items.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

130
131
132
133 **Code:**

```
def __init__(self, model_component: LayerModelComponent):
    if not isinstance(model_component, LayerModelComponent):
        raise TypeError("'model_component' not of type LayerMode
super().__init__(model_component)
```

layer  **Code:**

layer

Returns the map layer from the underlying model component object.

Returns:

Type	Description
------	-------------

QgsMapLayer Map layer corresponding from the underlying model component.

user_defined  **Code:**

user_defined

Returns whether the model component is user-defined or default that is shipped together with the plugin.

Returns:**Type Description**

`bool` True if the model component is user-defined else False if its a default component.

`clone` `abstractmethod`

 **Code:**`clone()`

Creates a deep copied version of the model item.

Returns:**Type Description**

`ModelComponentItem` Cloned version of the model item containing all the properties as the source.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py`

204
205
206
207
208
209
210
211
212

 **Code:**

```
@abstractmethod
def clone(self) -> "ModelComponentItemType":
    """Creates a deep copied version of the model item.

    :returns: Cloned version of the model item containing all
    the properties as the source.
    :rtype: ModelComponentItem
    """
    pass
```

```
create abstractmethod staticmethod

Code:

create(model_component)
```

Factory method for creating an instance of a model item.

This is an abstract method that needs to be implemented by subclasses.

Parameters:

Name	Type	Description	Default
model_component	BaseModelComponent	Source model component for creating the required corresponding item.	

Returns:

Type	Description
ModelComponentItem	Model component item for use in a standard item model.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202
```

 **Code:**

```
@staticmethod  
@abstractmethod  
def create(model_component: BaseModelComponent) -> "ModelComponentItem"  
    """Factory method for creating an instance of a model item.  
  
    This is an abstract method that needs to be implemented by  
    subclasses.  
  
    :param model_component: Source model component for creating the  
    corresponding item.  
    :type model_component: BaseModelComponent  
  
    :returns: Model component item for use in a standard item model.  
    :rtype: ModelComponentItem  
    """  
    pass
```

is_valid

```
is_valid()
```

Checks whether the map layer of the underlying model component object is valid.

Returns:**Type Description**

bool True if the map layer is valid, else False if map layer is invalid or of None type.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146
```

 **Code:**

```
def is_valid(self) -> bool:  
    """Checks whether the map layer of the underlying model  
    component object is valid.  
  
    :returns: True if the map layer is valid, else False  
    if map layer is invalid or of None type.  
    :rtype: bool  
    """  
  
    if self._model_component is None:  
        return False  
  
    return self._model_component.is_valid()
```

set_layer

 **Code:**

```
set_layer(layer)
```

Set the map layer for the component item.

It sets the :py:attr: ~path attribute of the underlying data model.

Parameters:

Name	Type	Description	Default
layer	QgsMapLayer	Map layer for the component item.required	

Returns:**Type** **Description**

bool Returns True if the layer was successfully set, else False if the layer is invalid.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

159
160
161 **Code:**
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185

```
def set_layer(self, layer: QgsMapLayer) -> bool:  
    """Set the map layer for the component item.  
  
    It sets the :py:attr:`~path` attribute of the  
    underlying data model.  
  
    :param layer: Map layer for the component item.  
    :type layer: QgsMapLayer  
  
    :returns: Returns True if the layer was successfully  
    set, else False if the layer is invalid.  
    :rtype: bool  
    """  
  
    if not layer:  
        return False  
  
    if not layer.isValid():  
        return False  
  
    path = layer.source()  
    self._model_component.path = path  
    if isinstance(layer, QgsRasterLayer):  
        self._model_component.layer_type = LayerType.RASTER  
    elif isinstance(layer, QgsVectorLayer):  
        self._model_component.layer_type = LayerType.VECTOR  
  
    return True
```

LayerItem

Bases: `QStandardItem`

Contains a custom identifier for an item used to define a layer for an implementation model.

`type`

 **Code:**

`type()`

Returns the type of the standard item.

Returns:

Type Description

- `int` Type identifier of the standard item.
- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py`

629
630
631
632
633
634
635

 **Code:**

```
def type(self) -> int:
    """Returns the type of the standard item.

    :returns: Type identifier of the standard item.
    :rtype: int
    """
    return LAYER_ITEM_TYPE
```

ModelComponentItem**Code:**

```
ModelComponentItem(model_component)
```

Bases: QStandardItem

Base standard item for a BaseModelComponent object.

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py](#)

44
45
46
47
48
49
50

```
def __init__(self, model_component: BaseModelComponent):  
    super().__init__(model_component.name)  
    self.setToolTip(model_component.name)  
  
    self._model_component = model_component  
    if self._model_component is not None:  
        self.update(self._model_component)
```

description **property**

Code:

```
description
```

Returns the description of the item.

Returns:**Type Description**

`str` Description of the item.

`model_component` `property`

 **Code:**

`model_component`

Returns an instance of the underlying model component object.

Returns:**Type Description**

`BaseModelComponent` Instance of underlying model component object.

`uuid` `property`

 **Code:**

`uuid`

Returns the UUID of the item.

Returns:**Type Description**

`str` UUID string of the item.

`clone` `abstractmethod`

 **Code:**

`clone()`

Creates a deep copied version of the model item.

Returns:

Type	Description
ModelComponentItem	Cloned version of the model item containing all the properties as the source.
• Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py	

```
111
112     🧑‍💻 Code:
113
114
115
116
117
118 @abstractmethod
119 def clone(self) -> "ModelComponentItemType":
    """Creates a deep copied version of the model item.
       :returns: Cloned version of the model item containing all
       the properties as the source.
       :rtype: ModelComponentItem
       """
    pass
```

create abstractmethod staticmethod

🧑‍💻 **Code:**

```
create(model_component)
```

Factory method for creating an instance of a model item.

This is an abstract method that needs to be implemented by subclasses.

Parameters:

Name	Type	Description	Default
model_component	BaseModelComponent	Source model component for creating the required corresponding item.	

Returns:

Type	Description
<code>ModelComponentItem</code>	Model component item for use in a standard item model.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py`

```
94
95  Code:
96
97
98
99
100
101 @staticmethod
102 @abstractmethod
103 def create(model_component: BaseModelComponent) -> "ModelComponentItem":
104     """Factory method for creating an instance of a model item.
105
106     This is an abstract method that needs to be implemented by
107     subclasses.
108
109     :param model_component: Source model component for creating the
110     corresponding item.
111     :type model_component: BaseModelComponent
112
113     :returns: Model component item for use in a standard item model.
114     :rtype: ModelComponentItem
115     """
116
117     pass
```

update

 **Code:**`update(model_component)`

Update the component-related properties of the item.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
52
53      Code:
54
55
56
57
58
59     def update(self, model_component: BaseModelComponent):
60         """Update the component-related properties of the item."""
61         if model_component is None:
62             return
63
64
65         self._model_component = model_component
66         self.setText(model_component.name)
67         self.setToolTip(model_component.name)
```

NcsPathwayItem

 **Code:**

```
NcsPathwayItem(ncs)
```

Bases: [LayerComponentItem](#)

Standard item for an NCS pathway object.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

229
230
231
232

 **Code:**

```
def __init__(self, ncs: NcsPathway):  
    super().__init__(ncs)  
    self._ncs_pathway = ncs  
    self._parent = None
```

ncs_pathway **property**

 **Code:**

ncs_pathway

Returns an instance of the underlying NcsPathway object.

Returns:

Type	Description
------	-------------

NcsPathway The underlying NcsPathway model object.

parent **property**

 **Code:**

parent

Returns the parent ImplementationModelItem if specified.

Returns:**Type****Description**

ImplementationModelItem Returns the parent item if set when this item is mapped to an ImplementationModelItem.

clone

 **Code:**`clone()`

Creates a cloned version of this item.

- Source code in `/home/samwelij/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py`

271
272
273
274
275

 **Code:**

```
def clone(self) -> "NcsPathwayItem":  
    """Creates a cloned version of this item."""  
    ncs = clone_ncs_pathway(self.ncs_pathway)  
  
    return NcsPathwayItem(ncs)
```

`create` `staticmethod`

 **Code:**`create(ncs)`

Creates an instance of the NcsPathwayItem from the model object.

Returns:

Type	Description
NcsPathwayItem	An instance of the NcsPathway item to be used in a standard model.

- Source code in /home/samwel1/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

261
262
263
264
265
266
267
268
269

 **Code:**

```
@staticmethod
def create(ncs: NcsPathway) -> "NcsPathwayItem":
    """Creates an instance of the NcsPathwayItem from the model object.

    :returns: An instance of the NcsPathway item to be used in a standard
    model.
    :rtype: NcsPathwayItem
    """
    return NcsPathwayItem(ncs)
```

is_carbon_valid **Code:**

```
is_carbon_valid()
```

Returns the validity of the carbon layers in the underlying NCSPathway model object.

Returns:

Type	Description
bool	True if the carbon layers are valid, else False.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

289
290
291
292
293
294
295
296
297

 **Code:**

```
def is_carbon_valid(self) -> bool:  
    """Returns the validity of the carbon layers in the  
    underlying NCSPathway model object.  
  
    :returns: True if the carbon layers are valid, else  
    False.  
    :rtype: bool  
    """  
    return self.ncs_pathway.is_carbon_valid()
```

json_data

 **Code:**

```
json_data()
```

Creates a mapping of NCS pathway property names and their corresponding values.

Returns:

Type Description

`str` JSON representation of property name-value pairs for an NCS pathway object.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py`

```
277
278  Code:
279
280
281
282
283
284
285     def json_data(self) -> str:
286         """Creates a mapping of NCS pathway property names
287         and their corresponding values.
288
289         :returns: JSON representation of property name-value
290         pairs for an NCS pathway object.
291         :rtype: str
292
293         """
294
295         ncs_attrs = ncs_pathway_to_dict(self._ncs_pathway)
296
297
298         return json.dumps(ncs_attrs)
```

setEnabled

Code :

`setEnabled(enabled)`

Override for default implementation that also enables or disables selection of the item.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

299
300
301
302
303
304

 **Code:**

```
def setEnabled(self, enabled: bool):
    """Override for default implementation that also
    enables or disables selection of the item.
    """
    self.setSelectable(enabled)
    super().setEnabled(enabled)
```

type

 **Code:**

```
type()
```

Returns the type of the standard item.

Returns:**Type Description**

int Type identifier of the standard item.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
253  
254  
255  
256  
257  
258  
259
```

 **Code:**

```
def type(self) -> int:  
    """Returns the type of the standard item.  
  
    :returns: Type identifier of the standard item.  
    :rtype: int  
    """  
  
    return NCS_PATHWAY_TYPE
```

NcsPathwayItemModel

Bases: ComponentItemModel

View model for NCS pathways.

add_ncs_pathway

 **Code:**

```
add_ncs_pathway(ncs)
```

Add an NCS pathway object to the model.

Parameters:

Name	Type	Description	Default
ncs	NcsPathway	NCS pathway object to the added to the view.required	

Returns:

Type Description

bool	True if the NCS pathway object was added successfully, else False if the NcsPathway object is invalid.
------	--

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
9999
```

 **Code:**

```
def add_ncs_pathway(self, ncs: NcsPathway) -> bool:  
    """Add an NCS pathway object to the model.  
  
    :param ncs: NCS pathway object to be added to the view.  
    :type ncs: NcsPathway  
  
    :returns: True if the NCS pathway object was added successfully,  
    else False if the NcsPathway object is invalid.  
    :rtype: bool  
    """  
  
    ncs_item = NcsPathwayItem.create(ncs)  
    self._update_display(ncs_item)  
  
    status = self.add_component_item(ncs_item)  
  
    self.sort(0)  
    self._re_index_rows()  
  
    return status
```

mimeData **Code:**

```
mimeData(indexes)
```

Serializes the NCS items corresponding to the specified indexes.

Parameters:

Name	Type	Description	Default
indexes	List[QModelIndex]	NCS items stored in the specified indexes.	required

Returns:

Type	Description
QtCore.QMimeData	Mime object containing serialized NCS items.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

932
933
934  **Code:**
935
936
937
938
939 def mimeData(self, indexes: typing.List[QtCore.QModelIndex]) -> QtCore.QMimeDat
940 """Serializes the NCS items corresponding to the specified indexes.
941
942
943 :param indexes: NCS items stored in the specified indexes.
944 :type indexes: list
945
946 :returns: Mime object containing serialized NCS items.
947 :rtype: QtCore.QMimeData
948 """
949 mime_data = QtCore.QMimeData()
950 item_data = QtCore.QByteArray()
951 data_stream = QtCore.QDataStream(item_data, QtCore.QIODevice.WriteOnly)
952
953
954 for idx in indexes:
955 if not idx.isValid():
956 continue
957
958 ncs_item = self.itemFromIndex(idx)
959 if ncs_item is None:
960 continue
961
962 # Do not add disabled items (e.g. disabled default NCS pathway items)
963 if not ncs_item.isEnabled():
964 continue
965
966 ncs_data = QtCore.QByteArray()
967 ncs_data.append(ncs_item.json_data())
968 data_stream << ncs_data
969
970 mime_data.setData(NCS_MIME_TYPE, item_data)

```
    return mime_data
```

mimeTypes

 **Code:**

```
mimeTypes()
```

Returns supported MIME types that can be used to describe a list of model indexes for NCS pathway items.

Returns:**Type Description**

`list` MIME type for NCS pathway items which is JSON string but MIME type is the default datalist type for Qt since it does not allow custom types.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py`

```
921  
922  
923  
924  
925  
926  
927  
928  
929  
930
```

 **Code:**

```
def mimeTypes(self) -> typing.List[str]:  
    """Returns supported MIME types that can be used to  
    describe a list of model indexes for NCS pathway items.  
  
    :returns: MIME type for NCS pathway items which is JSON  
    string but MIME type is the default datalist type for Qt  
    since it does not allow custom types.  
    :rtype: list  
    """  
    return [NCS_MIME_TYPE]
```

pathways

 **Code:**

```
pathways(valid_only=False)
```

Returns NCS pathway objects in the model.

Parameters:

Name	Type	Description	Default
valid_only	bool	Whether to only return NCS pathway objects that are valid.	False

Returns:

Type Description

list	All NCS pathway objects in the model (default), else only those NCS pathway objects that are valid if valid_only is True.
------	---

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
881
882      Code:
883
884
885
886
887
888     def pathways(self, valid_only: bool = False) -> typing.List[NcsPathway]:
889         """Returns NCS pathway objects in the model.
890
891         :param valid_only: Whether to only return NCS pathway objects
892             that are valid.
893         :type valid_only: bool
894
895
896         :returns: All NCS pathway objects in the model (default), else only
897             those NCS pathway objects that are valid if valid_only is True.
898         :rtype: list
899
900         """
901
902         ncs_pathways = self.model_components()
903
904
905         if valid_only:
906             return [p for p in ncs_pathways if p.is_valid()]
907
908
909         return ncs_pathways
```

remove_ncs_pathway

 **Code:**

```
remove_ncs_pathway(ncs_uuid)
```

Remove an NCS pathway item from the model.

param uuid: UUID of the NCS pathway item to be removed.

Returns:**Type Description**

`bool` True if the NCS pathway item was successfully removed, else False if there was not matching UUID.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py`

```
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909
```

 **Code:**

```
def remove_ncs_pathway(self, ncs_uuid: str) -> bool:  
    """Remove an NCS pathway item from the model.  
  
    param uuid: UUID of the NCS pathway item to be removed.  
    :type ncs_uuid: str  
  
    :returns: True if the NCS pathway item was successfully removed,  
    else False if there was not matching UUID.  
    :rtype: bool  
    """  
  
    return self.remove_component_item(ncs_uuid)
```

`supportedDropActions`

 **Code:**

```
supportedDropActions()
```

Configure the model to only support copying items in a drag-and-drop operation.

Returns:**Type****Description**

`QtCore.Qt.DropActions` Supported drag-and-drop action for NCS pathway items.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

```
911  
912  
913  
914  
915  
916  
917  
918  
919
```

 **Code:**

```
def supportedDropActions(self) -> QtCore.Qt.DropActions:  
    """Configure the model to only support copying items in a  
    drag-and-drop operation.  
  
    :returns: Supported drag-and-drop action for NCS pathway  
    items.  
    :rtype: QtCore.Qt.DropActions  
    """  
    return QtCore.Qt.CopyAction
```

update_ncs_pathway

 **Code:**

```
update_ncs_pathway(ncs)
```

Updates the NCS pathway item in the model.

Parameters:

Name	Type	Description	Default
ncs	NcsPathway	NcsPathway whose corresponding item is to be updated.required	

Returns:

Type	Description
bool	Returns True if the operation was successful else False if the matching item was not found in the model.

bool Returns True if the operation was successful else False if the matching item was not found in the model.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/component_item_model.py

842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864

 **Code:**

```
def update_ncs_pathway(self, ncs: NcsPathway) -> bool:
    """Updates the NCS pathway item in the model.

    :param ncs: NcsPathway whose corresponding item is to be updated.
    :type ncs: NcsPathway

    :returns: Returns True if the operation was successful else False
    if the matching item was not found in the model.
    """
    item = self.component_item_by_uuid(str(ncs.uuid))
    if item is None:
        return False

    status = self.update_item(item)
    if not status:
        return False

    self._update_display(item)

    self.sort(0)
    self._re_index_rows()

    return True
```

Implementation model editor

Dialog for creating or editing an implementation model.

ImplementationModelEditorDialog

 **Code:**

```
ImplementationModelEditorDialog(  
    parent=None,  
    implementation_model=None,  
    excluded_names=None,  
)
```

Bases: `QDialog`, `WidgetUi`

Dialog for creating or editing an implementation model entry.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_editor_dialog.py`

 **Code:**

```
45
46
47
48
49
50
51
52 def __init__(self, parent=None, implementation_model=None, excluded_names=
53     super().__init__(parent)
54     self.setupUi(self)
55
56
57
58 QgsGui.enableAutoGeometryRestore(self)
59
60
61 self._message_bar = QgsMessageBar()
62 self.vl_notification.addWidget(self._message_bar)
63
64
65 self.style_btn.setSymbolType(Qgis.SymbolType.Fill)
66
67
68
69 self.btn_color_ramp.setShowNull(False)
70 self.btn_color_ramp.setRandomColorRamp()
71 self.btn_color_ramp.setColorRampDialogTitle(
72     self.tr("Set Color Ramp for Output Implementation Model"))
73
74
75
76
77
78 self.buttonBox.accepted.connect(self._on_accepted)
79 self.btn_select_file.clicked.connect(self._on_select_file)
80 self.btn_help.clicked.connect(self.open_help)
81
82
83
84
85 icon_pixmap = QtGui.QPixmap(ICONS_PATH)
86 self.icon_la.setPixmap(icon_pixmap)
87
88
89 self.cbo_layer.setFilters(QgsMapLayerProxyModel.Filter.RasterLayer)
90
91
92 self._edit_mode = False
93 self._layer = None
94
95
96 self._excluded_names = excluded_names
97 if excluded_names is None:
```

```
self._excluded_names = []  
  
self._implementation_model = implementation_model  
if self._implementation_model is not None:  
    self._edit_mode = True  
    self._layer = self._implementation_model.to_map_layer()  
    self._update_controls()  
  
help_icon = FileUtils.get_icon("mActionHelpContents.svg")  
self.btn_help.setIcon(help_icon)
```

edit_mode property

 **Code:**

edit_mode

Returns the state of the editor.

Returns:

Type Description

bool True if the editor is editing an existing ImplementationModel object, else False if its creating a new object.

implementation_model property

 **Code:**

implementation_model

Returns a reference to the ImplementationModel object.

Returns:

Type Description

ImplementationModel Reference to the ImplementationModel object.

layer [property](#)**Code:****layer**

Returns the raster layer specified by the user, either existing layers in the map canvas or from the selected file.

Returns:

Type	Description
QgsRasterLayer	The raster layer specified by the user or None if not set.

[open_help](#)**Code:****open_help(activated)**

Opens the user documentation for the plugin in a browser.

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_editor_dialog.py](#)

311
312
313**Code:**

```
def open_help(self, activated: bool):
    """Opens the user documentation for the plugin in a browser."""
    open_documentation(USER_DOCUMENTATION_SITE)
```

output_layer_color_ramp

Code:

```
output_layer_color_ramp()
```

Returns the selected color ramp.

Returns:

Type	Description
------	-------------

<code>QgsColorRamp</code>	The color ramp selected by the user.
---------------------------	--------------------------------------

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_editor_dialog.py`

```
301  
302  
303  
304  
305  
306  
307  
308  
309
```

Code:

```
def output_layer_color_ramp(self) -> QgsColorRamp:  
    """Returns the selected color ramp.  
  
    :returns: The color ramp selected by the user.  
    :rtype: QgsColorRamp  
    """  
  
    color_ramp = self.btn_color_ramp.colorRamp()  
  
    return color_ramp
```

scenario_fill_symbol_layer

Code:

```
scenario_fill_symbol_layer()
```

Gets the first fill symbol layer in the symbol as set in the button.

It checks to ensure that there is at least one fill symbol layer contained in the symbol.

Returns:

Type	Description
<code>QgsFillSymbolLayer</code>	Fill symbol layer to be used in the implementation model.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_editor_dialog.py

```
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299
```

 **Code:**

```
def scenario_fill_symbol_layer(self) -> QgsFillSymbolLayer:  
    """Gets the first fill symbol layer in the symbol as  
    set in the button.  
  
    It checks to ensure that there is at least one fill symbol  
    layer contained in the symbol.  
  
    :returns: Fill symbol layer to be used in the implementation  
    model.  
    :rtype: QgsFillSymbolLayer  
    """  
  
    fill_symbol_layer = None  
    btn_symbol = self.style_btn.symbol()  
  
    for i in range(btn_symbol.symbolLayerCount()):  
        symbol_layer = btn_symbol.symbolLayer(i)  
        if isinstance(symbol_layer, QgsFillSymbolLayer):  
            fill_symbol_layer = symbol_layer  
            break  
  
    return fill_symbol_layer
```

validate **Code:**`validate()`

Validates if name has been specified.

Returns:**Type Description**

True True if the name have been set.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_editor_dialog.py`

Code:

```
179 def validate(self) -> bool:
180     """Validates if name has been specified.
181
182     :returns: True if the name have been set.
183     :rtype: True
184     """
185
186     status = True
187
188     self._message_bar.clearWidgets()
189
190     name = self.txt_name.text()
191     if not name:
192         msg = tr("Implementation model name cannot be empty.")
193         self._show_warning_message(msg)
194         status = False
195
196     if name.lower() in self._excluded_names:
197         msg = tr("name has already been used.")
198         self._show_warning_message(f'''{name}' {msg}''')
199         status = False
200
201     if not self.txt_description.toPlainText():
202         msg = tr("Description cannot be empty.")
203         self._show_warning_message(msg)
204         status = False
205
206     layer = self._get_selected_map_layer()
207     if layer and not layer.isValid():
208         msg = tr("Map layer is not valid.")
209         self._show_warning_message(msg)
210         status = False
```

```
fill_symbol_layer = self.scenario_fill_symbol_layer()
if fill_symbol_layer is None:
    msg = tr("No fill symbol defined for the scenario layer.")
    self._show_warning_message(msg)
    status = False

output_model_color_ramp = self.btn_color_ramp.colorRamp()
if output_model_color_ramp is None:
    msg = tr("No color ramp defined for the output model layer")
    self._show_warning_message(msg)
    status = False

return status
```

Implementation model widget

Container widget for configuring the implementation widget.

ImplementationModelContainerWidget



Code:

```
ImplementationModelContainerWidget(  
    parent=None, message_bar=None  
)
```

Bases: `QWidget`, `WidgetUi`

Widget for configuring the implementation model.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_widget.py`



Code :

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

Code:

```
def __init__(self, parent: QtWidgets.QWidget = None, message_bar: QgsMessageBar = None):
    super().__init__(parent)
    self.setupUi(self)

    self._message_bar = message_bar

    self._items_loaded = False

    self.btn_add_one.setIcon(FileUtils.get_icon("cplus_right_arrow.svg"))
    self.btn_add_one.setToolTip(self.tr("Add selected NCS pathway"))
    self.btn_add_one.clicked.connect(self._on_add_ncs_pathway)

    self.btn_add_all.setIcon(FileUtils.get_icon("cplus_double_right_arrows.svg"))
    self.btn_add_all.setToolTip(self.tr("Add all NCS pathways"))
    self.btn_add_all.clicked.connect(self._on_add_all_ncs_pathways)

    # NCS pathway view
    self.ncs_pathway_view = NcsComponentWidget()
    self.ncs_pathway_view.title = self.tr("NCS Pathways")
    self.ncs_layout.addWidget(self.ncs_pathway_view)

    # Implementation model view
    self.implementation_model_view = ImplementationModelComponentWidget()
    self.ipm_layout.addWidget(self.implementation_model_view)
    self.implementation_model_view.title = self.tr("Implementation Models")

    settings_manager.settings_updated[str, object].connect(self.on_settings_change)
    self.ncs_pathway_view.ncs_pathway_updated.connect(self.on_ncs_pathway_updated)
    self.ncs_pathway_view.ncs_pathway_removed.connect(self.on_ncs_pathway_removed)
```

```
    self.ncs_pathway_view.items_reloaded.connect(self._on_ncs_pathways_reloaded)

    self.load()
```

enable_default_items

Code:

```
enable_default_items(enable)
```

Enable or disable default NCS pathway and implementation model items.

Parameters:

Name	Type	Description	Default
enable	bool	True to enable or False to disable default items.required	
• Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_widget.py			

```
137
138     def enable_default_items(self, enable: bool):
139         """Enable or disable default NCS pathway and implementation model items.
140
141         :param enable: True to enable or False to disable default items.
142         :type enable: bool
143
144         """
145
146         if not self._items_loaded:
147             return
148
149
150         self.ncs_pathway_view.enable_default_items(enable)
151         self.implementation_model_view.enable_default_items(enable)
```

implementation_models

Code:

```
implementation_models()
```

Returns the user-defined implementation models in the Implementation Models view.

Returns:

Type Description

`list` User-defined implementation models for the current scenario.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_widget.py`

```
93
94
95
96
97
98
99
100
```

```
def implementation_models(self) -> typing.List[ImplementationModel]:
    """Returns the user-defined implementation models in the
    Implementation Models view.

    :returns: User-defined implementation models for the current scenario.
    :rtype: list
    """
    return self.implementation_model_view.models()
```

is_valid

Code:

```
is_valid()
```

Check if the user input is valid.

This checks if there is one implementation model defined with at least one NCS pathway under it.

Returns:

Type Description

bool True if the implementation model configuration is valid, else False at least until there is one implementation model defined with at least one NCS pathway under it.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_widget.py`

178

179

180

181

182

183

184

185

186

 **Code:**

187

188

189

190

191

192

193

194

195

196

197

198

199

```
def is_valid(self) -> bool:  
    """Check if the user input is valid.  
  
    This checks if there is one implementation model defined with at  
    least one NCS pathway under it.  
  
    :returns: True if the implementation model configuration is  
    valid, else False at least until there is one implementation  
    model defined with at least one NCS pathway under it.  
    :rtype: bool  
    """  
  
    implementation_models = self.implementation_models()  
    if len(implementation_models) == 0:  
        return False  
  
    status = False  
    for im in implementation_models:  
        if len(im.pathways) > 0 or im.to_map_layer() is not None:  
            status = True  
            break  
  
    return status
```

load **Code:**`load()`

Load NCS pathways and implementation models to the views.

This function is idempotent as items will only be loaded once on initial call.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_widget.py`

```
74
75  Code:
76
77
78
79
80
81 def load(self):
82     """Load NCS pathways and implementation models to the views.
83
84     This function is idempotent as items will only be loaded once
85     on initial call.
86     """
87
88     if not self._items_loaded:
89         self.ncs_pathway_view.load()
90         self.implementation_model_view.load()
91         self._items_loaded = True
```

ncs_pathways **Code:**`ncs_pathways()`

Gets the NCS pathway objects in the NCS Pathways view.

Returns:

Type Description

`list` NCS pathway objects, both default and user-defined.

• Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_widget.py`

```
85
86      Code:
87
88
89
90
91
def ncs_pathways(self) -> typing.List[NcsPathway]:
    """Gets the NCS pathway objects in the NCS Pathways view.

    :returns: NCS pathway objects, both default and user-defined.
    :rtype: list
    """
    return self.ncs_pathway_view.pathways()
```

`on_ncs_pathway_removed`

 **Code:**

```
on_ncs_pathway_removed(ncs_pathway_uuid)
```

Slot raised when an NCS pathway has been removed.

Parameters:

Name	Type	Description	Default
<code>ncs_pathway_uuid</code>	<code>str</code>	Unique identifier of the removed NCS pathway item.	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_widget.py

```
129
130     def on_ncs_pathway_removed(self, ncs_pathway_uuid: str):
131         """Slot raised when an NCS pathway has been removed.
132
133         :param ncs_pathway_uuid: Unique identified of the removed NCS pathway item.
134         :type ncs_pathway_uuid: str
135
136         self.implementation_model_view.remove_ncs_pathway_items(ncs_pathway_uuid)
```

on_ncs_pathway_updated

 **Code:**

```
on_ncs_pathway_updated(ncs_pathway)
```

Slot raised when an NCS pathway has been updated.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_widget.py

```
125
126     def on_ncs_pathway_updated(self, ncs_pathway: NcsPathway):
127         """Slot raised when an NCS pathway has been updated."""
128         self.implementation_model_view.update_ncs_pathway_items(ncs_pathway)
```

on_settings_changed

Code:

```
on_settings_changed(name, value)
```

Slot raised when settings has been changed.

Parameters:

Name	Type	Description	Default
name	str	Name of the setting that has changed. required	
value	Any	New value for the given settings name.required	

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_widget.py](#)

```
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176
```

```
 Code:  
  
def on_settings_changed(self, name: str, value: typing.Any):  
    """Slot raised when settings has been changed.  
  
    :param name: Name of the setting that has changed.  
    :type name: str  
  
    :param value: New value for the given settings name.  
    :type value: Any  
    """  
  
    # Update the NCS pathway and carbon layer paths when  
    # BASE_DIR has been updated.  
    if name == Settings.BASE_DIR.value:  
        self.ncs_pathway_view.load()
```

selected_im_items

Code:

```
selected_im_items()
```

Returns the currently selected instances of ImplementationModelItem.

If an item is disabled then it will be excluded from the selection.

Returns:

Type Description

- `list` Currently selected instances of ImplementationModelItem or an empty list if there is no selection of IM items.
- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_widget.py`

Code:

```
226
227
228
229
230
231
232
233     def selected_im_items(self) -> typing.List[ImplementationModelItem]:
234         """Returns the currently selected instances of ImplementationModelItem.
235
236         If an item is disabled then it will be excluded from the selection.
237
238         :returns: Currently selected instances of ImplementationModelItem or
239         an empty list if there is no selection of IM items.
240         :rtype: list
241
242         """
243         return [
244             item
245             for item in self.selected_items()
246             if isinstance(item, ImplementationModelItem)
247         ]
```

`selected_items` **Code:**`selected_items()`

Returns the selected model component item types which could be NCS pathway or implementation model items.

If an item is disabled then it will be excluded from the selection.

These are cloned objects so as not to interfere with the underlying data models when used for scenario analysis. Otherwise, one can also use the data models from the MVC item model.

Returns:**Type Description**`list` Selected model component items.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_widget.py

201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224

 **Code:**

```
def selected_items(self) -> typing.List[ModelComponentItemType]:
    """Returns the selected model component item types which could be
    NCS pathway or implementation model items.

    If an item is disabled then it will be excluded from the
    selection.

    These are cloned objects so as not to interfere with the
    underlying data models when used for scenario analysis. Otherwise,
    one can also use the data models from the MVC item model.

    :returns: Selected model component items.
    :rtype: list
    """
    ref_items = self.implementation_model_view.selected_items()
    cloned_items = []
    for ref_item in ref_items:
        if not ref_item.isEnabled():
            continue

        clone_item = ref_item.clone()
        cloned_items.append(clone_item)

    return cloned_items
```

show_message **Code:**

```
show_message(message, level=Qgis.Warning)
```

Shows message if message bar has been specified.

Parameters:

Name	Type	Description	Default
message	str	Text to display in the message bar.required	
level	Qgis.MessageLevel	Message level type	Warning

- Source code in /home/samwel/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/implementation_model_widget.py

149
150
151
152
153
154
155
156
157
158
159
160
161
162

```
def show_message(self, message, level=Qgis.Warning):
    """Shows message if message bar has been specified.

    :param message: Text to display in the message bar.
    :type message: str

    :param level: Message level type
    :type level: Qgis.MessageLevel
    """

    if self._message_bar is None:
        return

    self._message_bar.clearWidgets()
    self._message_bar.pushMessage(message, level=level)
```

Items selection dialog

Item selection dialog file

ItemsSelectionDialog

 **Code:**

```
ItemsSelectionDialog(  
    parent,  
    parent_item=None,  
    items=[],  
    item_type=ImplementationModel,  
)
```

Bases: `QDialog`, `DialogUi`

Dialog for handling items selection

Constructor

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/items_selection_dialog.py`

30
31  **Code:**
32
33
34
35
36
37 def __init__(
38 self, parent, parent_item=None, items=[], item_type=ImplementationMod
39):
40 """Constructor"""
41 super().__init__()
42 self.setupUi(self)
43 self.parent = parent
44 self.parent_item = parent_item
45
46 self.item_type = item_type
47 self.items = items
48
49
50
51 select_all_btn = QtWidgets.QPushButton(tr("Select All"))
52 select_all_btn.setToolTip(tr("Select the all listed items"))
53 select_all_btn.clicked.connect(self.select_all_clicked)
54 self.mButtonBox.addButton(select_all_btn, QtWidgets.QDialogButtonBox.
55
56
57 clear_all_btn = QtWidgets.QPushButton(tr("Clear Selection"))
58 clear_all_btn.setToolTip(tr("Clear the current selection"))
59 clear_all_btn.clicked.connect(self.clear_all_clicked)
60 self.mButtonBox.addButton(clear_all_btn, QtWidgets.QDialogButtonBox.A
61
62
63
64 toggle_selection_btn = QtWidgets.QPushButton(tr("Toggle Selection"))
65 toggle_selection_btn.clicked.connect(self.toggle_selection_clicked)
66 self.mButtonBox.addButton(
67 toggle_selection_btn, QtWidgets.QDialogButtonBox.ActionRole
68)
69
70
71
72
73
74
75
76
77
78
79
80
81 self.mButtonBox.accepted.connect(self.accept)
82
83
84
85
86 self.set_items()

```
87
88
89
90
91
92
for index in range(self.list_widget.count()):
    item = self.list_widget.item(index)
    item_uuid = item.data(QtCore.Qt.UserRole)

    if self.item_type is ImplementationModel:
        model = settings_manager.get_implementation_model(str(item_uuid))

        layer_model_uuids = [item.uuid for item in self.items]
        model_layer_uuids = [
            layer.get("uuid")
            for layer in model.priority_layers
            if layer is not None
        ]

        if (
            self.parent_item is not None
            and str(self.parent_item.get("uuid")) in model_layer_uuids
            or (model.uuid in layer_model_uuids):
            item.setCheckState(QtCore.Qt.Checked)
        else:
            layer = settings_manager.get_priority_layer(str(item_uuid))
            group_uuids = []

            for group in layer.get("groups"):
                group = settings_manager.find_group_by_name(group.get("name"))

                if group is not None:
                    group_uuids.append(str(group.get("uuid")))

            if self.parent_item.get("uuid") in group_uuids:
                item.setCheckState(QtCore.Qt.Checked)
```

accept **Code:**`accept()`

Saves the item selection

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/items_selection_dialog.py`

169
170
171
172

 **Code:**

```
def accept(self):
    """Saves the item selection"""
    self.parent.set_selected_items(self.selected_items(), self.ui)
    super().accept()
```

clear_all_clicked **Code:**`clear_all_clicked()`

Slot for handling clear fselection for all items.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/items_selection_dialog.py

180
181
182
183
184

 **Code:**

```
def clear_all_clicked(self):
    """Slot for handling clear fselection for all items."""
    for item_index in range(self.list_widget.count()):
        item_item = self.list_widget.item(item_index)
        item_item.setCheckState(QtCore.Qt.Unchecked)
```

select_all_clicked

 **Code:**

```
select_all_clicked()
```

Slot for handling selection for all items.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/items_selection_dialog.py

174
175
176
177
178

 **Code:**

```
def select_all_clicked(self):
    """Slot for handling selection for all items."""
    for item_index in range(self.list_widget.count()):
        item_item = self.list_widget.item(item_index)
        item_item.setCheckState(QtCore.Qt.Checked)
```

selected_items

 **Code:**

```
selected_items()
```

Returns the selected items from the dialog

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/items_selection_dialog.py

```
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142
```

 **Code:**

```
def selected_items(self):  
    """Returns the selected items from the dialog"""  
    if self.item_type is ImplementationModel:  
        items = settings_manager.get_all_implementation_models()  
    else:  
        all_layers = settings_manager.get_priority_layers()  
        items = []  
        for layer in all_layers:  
            model_layer = PriorityLayer(  
                uuid=uuid.UUID(layer.get("uuid")),  
                name=layer.get("name"),  
                description=layer.get("description"),  
                groups=layer.get("groups"),  
            )  
            items.append(model_layer)  
  
    items_text = []  
    for index in range(self.list_widget.count()):  
        item = self.list_widget.item(index)  
        if item.checkState() == QtCore.Qt.Checked:  
            items_text.append(item.text())  
    item_names = ",".join(items_text)  
    items = [item for item in items if item.name in item_names]  
    return items
```

set_items **Code:**`set_items()`

Sets the item list in the dialog

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/items_selection_dialog.py

94
95 **Code:**
96
97
98
99
100
101 def set_items(self):
102 """Sets the item list in the dialog"""
103 if self.item_type is ImplementationModel:
104 items = settings_manager.get_all_implementation_models()
105 else:
106 all_layers = settings_manager.get_priority_layers()
107 items = []
108 for layer in all_layers:
109 model_layer = PriorityLayer(
110 uuid=uuid.UUID(layer.get("uuid")),
111 name=layer.get("name"),
112 description=layer.get("description"),
113 groups=layer.get("groups"),
114)
115 items.append(model_layer)
116
117
118 for item in items:
119 list_widget_item = QtWidgets.QListWidgetItem(item.name)
120 list_widget_item.setFlags(
121 list_widget_item.flags() | QtCore.Qt.ItemIsUserCheckable
122)
123 list_widget_item.setData(QtCore.Qt.UserRole, item.uuid)
124 list_widget_item.setCheckState(QtCore.Qt.Unchecked)
125 self.list_widget.addItem(list_widget_item)

toggle_selection_clicked **Code:**`toggle_selection_clicked()`

Toggles all the current items selection.

- Source code in `/home/samwel1/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/items_selection_dialog.py`

186
187
188
189
190
191
192
193
194

```
def toggle_selection_clicked(self):
    """Toggles all the current items selection."""
    for item_index in range(self.list_widget.count()):
        item_item = self.list_widget.item(item_index)
        state = item_item.checkState()
        if state == QtCore.Qt.Checked:
            item_item.setCheckState(QtCore.Qt.Unchecked)
        elif state == QtCore.Qt.Unchecked:
            item_item.setCheckState(QtCore.Qt.Checked)
```

unselected_items **Code:**`unselected_items()`

Returns unselected items from the dialog

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/items_selection_dialog.py

```
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167
```

 **Code:**

```
def unselected_items(self):  
    """Returns unselected items from the dialog"""  
    if self.item_type is ImplementationModel:  
        items = settings_manager.get_all_implementation_models()  
    else:  
        all_layers = settings_manager.get_priority_layers()  
        items = []  
        for layer in all_layers:  
            model_layer = PriorityLayer(  
                uuid=uuid.UUID(layer.get("uuid")),  
                name=layer.get("name"),  
                description=layer.get("description"),  
                groups=layer.get("groups"),  
            )  
            items.append(model_layer)  
  
    items_text = []  
    for index in range(self.list_widget.count()):  
        item = self.list_widget.item(index)  
        if item.checkState() == QtCore.Qt.Unchecked:  
            items_text.append(item.text())  
    item_names = ",".join(items_text)  
    items = [item for item in items if item.name in item_names]  
    return items
```

Map repeat item widget

Widget for the custom CPLUS layout map item.

CplusMapLayoutItemGuiMetadata

 **Code:**

```
CplusMapLayoutItemGuiMetadata()
```

Bases: `QgsLayoutItemAbstractGuiMetadata`

GUI metadata for a CPLUS map layout item.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/map_repeat_item_widget.py`

55
56

 **Code:**

```
def __init__(self):
    super().__init__(CPLUS_MAP_REPEAT_ITEM_TYPE, CPLUS_ITEM_NAME)
```

createItem

 **Code:**

```
createItem(layout)
```

Factory override that returns the map item.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/map_repeat_item_widget.py

62
63
64 **Code:**

```
def createItem(self, layout) -> QgsLayoutItem:  
    """Factory override that returns the map item."""  
    return CplusMapRepeatItem(layout)
```

createItemWidget

 **Code:**

```
createItemWidget(item)
```

Factory override for the item widget.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/map_repeat_item_widget.py

58
59
60 **Code:**

```
def createItemWidget(self, item) -> QtWidgets.QWidget:  
    """Factory override for the item widget."""  
    return CplusMapRepeatItemWidget(None, item)
```

creationIcon

 **Code:**

```
creationIcon()
```

Factory override for item icon.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/map_repeat_item_widget.py`

70
71
72

 **Code:**

```
def creationIcon(self) -> QtGui.QIcon:  
    """Factory override for item icon."""  
    return FileUtils.get_icon("mLayoutItemMap_cplus_add.svg")
```

newItemAddedToLayout

 **Code:**

```
newItemAddedToLayout(map_repeat_item)
```

Define action that is called when the CplusMapItem is added to a layout through the GUI.

Parameters:

Name	Type	Description	Default
<code>map_repeat_item</code>	CplusMapRepeatItem	Map repeat item to be added to the layout.	required

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/map_repeat_item_widget.py

```
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93
```

 **Code:**

```
def newItemAddedToLayout(self, map_repeat_item: CplusMapRepeatItem):  
    """Define action that is called when the CplusMapItem  
    is added to a layout through the GUI.  
  
    :param map_repeat_item: Map repeat item to be added to the layout.  
    :type map_repeat_item: CplusMapRepeatItem  
    """  
  
    items = map_repeat_item.layout().items()  
    counter = 1  
    for item in items:  
        if isinstance(item, CplusMapRepeatItem):  
            counter += 1  
  
        # Set frame properties  
        map_repeat_item.setFrameEnabled(True)  
        map_repeat_item.setFrameJoinStyle(QtCore.Qt.MiterJoin)  
        map_repeat_item.setFrameStrokeColor(QtGui.QColor(132, 192, 68))  
        map_repeat_item.setFrameStrokeWidth(QgsLayoutMeasurement(0.4))  
  
    map_repeat_item.setId(f"{{CPLUS_ITEM_NAME}} {counter}s")
```

visibleName

 **Code:**

```
visibleName()
```

Override for user-visible name identifying the item.

- Source code in `/home/samwel/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/map_repeat_item_widget.py`

66
67
68

 **Code:**

```
def visibleName(self) -> str:  
    """Override for user-visible name identifying the item."""  
    return CPLUS_ITEM_NAME
```

CplusMapRepeatWidgetItem

 **Code:**

```
CplusMapRepeatWidgetItem(parent, layout_object)
```

Bases: `QgsLayoutItemBaseWidget`, `WidgetUi`

Widget for configuring the CPLUS layout map repeatitem.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/map_repeat_item_widget.py

36
37
38
39
40
41
42
43
44
45
46
47
48
49

 **Code:**

```
def __init__(self, parent, layout_object: CplusMapRepeatItem):
    super().__init__(parent, layout_object)
    self.setupUi(self)

    self._map_item = layout_object

    # Common item properties widget
    self._prop_widget = QgsLayoutItemPropertiesWidget(self, layout_object)
    self.layout.addWidget(self._prop_widget, 2, 0, 1, 2)

    self.cbo_map_type.addItem(
        self.tr("Implementation model"),
        ModelComponentType.IMPLEMENTATION_MODEL.value,
    )
```

Model component widget

Composite list view-based widgets for displaying implementation model and NCS pathway items.

ImplementationModelComponentWidget

 **Code:**

```
ImplementationModelComponentWidget(parent=None)
```

Bases: **ModelComponentWidget**

Widget for displaying and managing implementation models.

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py](#)

```
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504
```

 **Code:**

```
def __init__(self, parent=None):  
    super().__init__(parent)  
  
    self.item_model = IMItemModel(parent)  
    self.item_model.im_pathways_updated.connect(self.on_pathways_updated)  
  
    self.lst_model_items.setAcceptDrops(True)  
    self.lst_model_items.setDragDropMode(QtWidgets.QAbstractItemView.DropOnly)  
    self.lst_model_items.setDropIndicatorShown(True)  
  
    self.btn_reload.setVisible(False)  
  
    self.btn_pixel_editor = None  
  
    self.add_auxiliary_widgets()
```

add_auxiliary_widgets

Code:

```
add_auxiliary_widgets()
```

Adds additional action widgets for managing implementation models.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

516
517
518
519
520
521
522
523
524
525

```
def add_auxiliary_widgets(self):
    """Adds additional action widgets for managing implementation models."""
    self.btn_pixel_editor = QtWidgets.QToolButton(self)
    style_icon = FileUtils.get_icon("rendererCategorizedSymbol.svg")
    self.btn_pixel_editor.setIcon(style_icon)
    self.btn_pixel_editor.setToolTip(
        self.tr("Show dialog for ordering pixel values for styling."))
    self.btn_pixel_editor.clicked.connect(self.on_show_pixel_value_editor)
    self.add_action_widget(self.btn_pixel_editor)
```

add_implementation_model

Code:

```
add_implementation_model(implementation_model, layer=None)
```

Adds an implementation model object to the view with the option of specifying the layer.

Parameters:

Name	Type	Description	Default
implementation_model	ImplementationModel	Implementation model object to be added to the view.	required
layer	QgsMapLayer	Optional map layer to be added to the model.	None

Returns:

Type	Description
bool	True if the implementation model was successfully added, else False.

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py](#)

764
765 Code:
766
767
768
769
770
771 def add_implementation_model(
772 self, implementation_model: ImplementationModel, layer: QgsMapLayer = None
773):
774 """Adds an implementation model object to the view with the option of
775 specifying the layer.
776
777 :param implementation_model: Implementation model object
778 to be added to the view.
779 :type implementation_model: ImplementationModel
780
781 :param layer: Optional map layer to be added to the model.
782 :type layer: QgsMapLayer
783
784 :returns: True if the implementation model was successfully added, else
785 False.
786 :rtype: bool
787 """
788
789 return self.item_model.add_implementation_model(implementation_model, lay

add_ncs_pathway_items

 **Code:**

```
add_ncs_pathway_items(ncs_items)
```

Adds an NCS pathway item to the collection.

One, and only one, target implementation model item needs to have been selected.

Parameters:

Name	Type	Description	Default
ncs_items	List[NcsPathwayItem]	NCS pathway items to be added to the implementation model.	required

Returns:

Type	Description
bool	True if the item was successfully added, else False.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py`

728
729
730  **Code:**
731
732
733
734
735 def add_ncs_pathway_items(self, ncs_items: typing.List[NcsPathwayItem])

736 """Adds an NCS pathway item to the collection.
737
738 One, and only one, target implementation model item needs
739 to have been selected.
740
741 :param ncs_items: NCS pathway items to be added to the
742 implementation model.
743 :type ncs_items: list
744
745 :returns: True if the item was successfully added, else False.
746 :rtype: bool
747 """
748 selected_models = self.selected_items()
749 if len(selected_models) == 0 or len(selected_models) > 1:
750 return False
751
752 sel_model = selected_models[0]
753 item_type = sel_model.type()
754
755 # Use the parent to add the NCS item
756 if item_type == NCS_PATHWAY_TYPE:
757 if sel_model.parent is None:
758 return False
759
760 sel_model = sel_model.parent
761
762 elif item_type == LAYER_ITEM_TYPE:
763 return False
764
765 status = True

```
for ncs_item in ncs_items:  
    status = self.item_model.add_ncs_pathway(ncs_item, sel_model)  
  
return status
```

clear

 **Code:**

```
clear()
```

Removes all implementation model items in the view.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py`

572
573
574
575
576

```
def clear(self):  
    """Removes all implementation model items in the view."""  
    items = self.model_items()  
    for item in items:  
        self.item_model.remove_implementation_model(item.uuid)
```

load

 **Code:**

```
load()
```

Load implementation models from settings.

- Source code in /home/samwelij/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

```
578  
579  
580  
581  
582  
583
```

 **Code:**

```
def load(self):  
    """Load implementation models from settings."""  
    self.clear()  
  
    for imp_model in settings_manager.get_all_implementation_models():  
        self.add_implementation_model(imp_model)
```

model_items

```
model_items()
```

Returns a collection of all ImplementationModelItem objects in the list view.

Returns:

Type Description

list Collection of ImplementationModelItem objects in the list view.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

```
562  
563  
564     Code:  
565  
566  
567  
568  
569  
570  
  
def model_items(self) -> typing.List[ImplementationModelItem]:  
    """Returns a collection of all ImplementationModelItem objects  
    in the list view.  
  
    :returns: Collection of ImplementationModelItem objects in  
    the list view.  
    :rtype: list  
    """  
  
    return self.item_model.model_items()
```

model_names

Code:

```
model_names()
```

Gets the names of the implementation models in the item model.

Returns:

Type Description

list	Returns the names of implementation models in lower case or an empty list if the item model has not been set.
------	---

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py`

```
542
543
544
545
546
547
548
549
550     def model_names(self) -> typing.List[str]:
551         """Gets the names of the implementation models in the item model.
552
553         :returns: Returns the names of implementation models in lower
554         case or an empty list if the item model has not been set.
555         :rtype: list
556
557         """
558
559         if self._item_model is None:
560             return []
561
562
563         model_components = self._item_model.models()
564
565
566         return [mc.name.lower() for mc in model_components]
```

models

Code :

models()

Returns a collection of ImplementationModel objects in the list view.

Returns:

Type Description

list Collection of ImplementationModel objects in the list view.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

```
506
507      Code:
508
509
510
511
512
513
514
def models(self) -> typing.List[ImplementationModel]:
    """Returns a collection of ImplementationModel objects in the
    list view.

    :returns: Collection of ImplementationModel objects in the
    list view.
    :rtype: list
    """
    return self.item_model.models()
```

on_pathways_updated

```
      Code:
on_pathways_updated(im_item)
```

Slot raised when the pathways of an ImplementationModelItem have been added or removed. Persist this information in settings.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

```
556  
557  
558  
559  
560
```

 **Code:**

```
def on_pathways_updated(self, im_item: ImplementationModelItem):  
    """Slot raised when the pathways of an ImplementationModelItem  
    have been added or removed. Persist this information in settings.  
    """  
    self._save_item(im_item)
```

on_show_pixel_value_editor **Code:**

```
on_show_pixel_value_editor()
```

Slot raised to show editor dialog for managing IM pixel values for styling.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

```
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540
```

 **Code:**

```
def on_show_pixel_value_editor(self):  
    """Slot raised to show editor dialog for managing IM pixel values for styling.  
    pixel_dialog = PixelValueEditorDialog(self)  
    if pixel_dialog.exec_() == QtWidgets.QDialog.Accepted:  
        # Update pixel values  
        pixel_values = pixel_dialog.item_mapping  
        for val, im_id in pixel_values.items():  
            imp_model = settings_manager.get_implementation_model(im_id)  
            if not imp_model:  
                continue  
            imp_model.style_pixel_value = val  
            settings_manager.update_implementation_model(imp_model)  
  
        self.load()
```

reassign_pixel_values

 **Code:**

```
reassign_pixel_values(start_position)
```

Reassign the styling pixel values for implementation models from the given start position.

It is important to call this function when the maximum pixel value does not match the number of implementation models such as when one or more implementation models have been deleted.

Parameters:

Name	Type	Description	Default
start_position	int	Position to start reassigning the pixel values.required	

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

705
706
707 **Code:**
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726

```
def reassign_pixel_values(self, start_position: int):
    """Reassign the styling pixel values for implementation models
    from the given start position.

    It is important to call this function when the maximum pixel
    value does not match the number of implementation models such
    as when one or more implementation models have been deleted.

    :param start_position: Position to start reassigning the pixel
    values.
    :type start_position: int
    """
    sorted_models = sorted(
        settings_manager.get_all_implementation_models(),
        key=lambda model: model.style_pixel_value,
    )
    remap_models = sorted_models[start_position - 1 :]
    for val, imp_model in enumerate(remap_models, start=start_position):
        imp_model.style_pixel_value = val
        settings_manager.update_implementation_model(imp_model)

    self.load()
```

remove_ncs_pathway_items

Code:

```
remove_ncs_pathway_items(ncs_pathway_uuid)
```

Delete NCS pathway items used for IMs that are linked to the given NCS pathway.

Parameters:

Name	Type	Description	Default
ncs_pathway_uuid	str	NCS pathway whose corresponding items will be deleted in the required implementation model items that contain it.	

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py`

```
813  
814  
815  
816  
817  
818  
819  
820  
821
```

Code:

```
def remove_ncs_pathway_items(self, ncs_pathway_uuid: str):  
    """Delete NCS pathway items used for IMs that are linked to the  
    given NCS pathway.  
  
    :param ncs_pathway_uuid: NCS pathway whose corresponding items will be  
    deleted in the implementation model items that contain it.  
    :type ncs_pathway_uuid: str  
    """  
    self.item_model.remove_ncs_pathway_items(ncs_pathway_uuid)
```

update_ncs_pathway_items

 **Code:**

```
update_ncs_pathway_items(ncs_pathway)
```

Update NCS pathway items used for IMs that are linked to the given NCS pathway.

Parameters:

Name	Type	Description	Default
ncs_pathway	NcsPathway	NCS pathway whose attribute values will be updated for the required related pathways used in the IMs.	

Returns:

Type	Description
bool	True if NCS pathway items were updated, else False.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py`

800
801  **Code:**
802
803
804
805
806
807 def update_ncs_pathway_items(self, ncs_pathway: NcsPathway) -> bool:
808 """Update NCS pathway items used for IMs that are linked to the
809 given NCS pathway.
810
811 :param ncs_pathway: NCS pathway whose attribute values will be updated
812 for the related pathways used in the IMs.
813 :type ncs_pathway: NcsPathway
814
815 :returns: True if NCS pathway items were updated, else False.
816 :rtype: bool
817 """
818
819 return self.item_model.update_ncs_pathway_items(ncs_pathway)

ModelComponentWidget

 **Code:**

```
ModelComponentWidget(parent=None, item_model=None)
```

Bases: `QWidget`, `WidgetUi`

Widget for displaying and managing model items in a list view.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py`

46
47  **Code:**
48
49
50
51
52
53 def __init__(self, parent=None, item_model=None):
54 super().__init__(parent)
55 self.setupUi(self)
56
57 self._item_model = item_model
58 if self._item_model is not None:
59 self.item_model = self._item_model
60
61 self.lst_model_items.doubleClicked.connect(self._on_double_click_item)
62
63 add_icon = FileUtils.get_icon("symbologyAdd.svg")
64 self.btn_add.setIcon(add_icon)
65 self.btn_add.clicked.connect(self._on_add_item)
66
67 remove_icon = FileUtils.get_icon("symbologyRemove.svg")
68 self.btn_remove.setIcon(remove_icon)
69 self.btn_remove.setEnabled(False)
70 self.btn_remove.clicked.connect(self._on_remove_item)
71
72 edit_icon = FileUtils.get_icon("mActionToggleEditing.svg")
73 self.btn_edit.setIcon(edit_icon)
74 self.btn_edit.setEnabled(False)
75 self.btn_edit.clicked.connect(self._on_edit_item)
76
77 reload_icon = FileUtils.get_icon("mActionReload.svg")
78 self.btn_reload.setIcon(reload_icon)
 self.btn_reload.setToolTip(self.tr("Refresh view"))
 self.btn_reload.clicked.connect(self._on_reload)

 self.btn_edit_description.setIcon(edit_icon)
 self.btn_edit_description.setToolTip(self.tr("Edit description"))

```
self.btn_edit_description.setEnabled(False)
self.btn_edit_description.clicked.connect(self._on_update_description)
```

item_model property writable

 **Code:**

item_model

Returns the component item model for managing items the list view.

Returns:

Type **Description**

ComponentItemModel Component item model for managing items the list view.

selection_model property

 **Code:**

selection_model

Gets the item's view selection model.

Returns:

Type **Description**

QtCore.QItemSelectionModel The item's view selection model.

title property writable

 **Code:**

title

Returns the title of the view.

Returns:

Type Description

str Title of the view.

add_action_widget

 **Code:**

```
add_action_widget(widget)
```

Adds an auxiliary widget below the list view from the left-hand side.

Parameters:

Name	Type	Description	Default
widget	QWidget	Widget to be added to the collection of controls below the list view.required	

- Source code in [/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py](#)

325
326
327
328
329
330
331
332

 **Code:**

```
def add_action_widget(self, widget: QtWidgets.QWidget):
    """Adds an auxiliary widget below the list view from the left-hand side.

    :param widget: Widget to be added to the collection of controls
    below the list view.
    :type widget: QtWidgets.QWidget
    """
    self.widget_container.addWidget(widget)
```

clear

 **Code:**`clear()`

Remove all items in the view. To be implemented by subclasses.

- Source code in `/home/samwelij/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py`

279
280
281
282

 **Code:**

```
def clear(self):
    """Remove all items in the view. To be implemented
    by subclasses."""
    pass
```

clear_description

 **Code:**`clear_description()`

Clears the content in the description text box.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

195
196
197
198

 **Code:**

```
def clear_description(self):
    """Clears the content in the description text box."""
    self.txt_item_description.clear()
    self.txt_item_description.setToolTip("")
```

enable_default_items

 **Code:**

```
enable_default_items(state)
```

Enable or disable default model component items in the view.

Parameters:

Name	Type	Description	Default
state	bool	True to enable or False to disable default model component items.required	

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

```
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323
```

 **Code:**

```
def enable_default_items(self, state: bool):  
    """Enable or disable default model component items in the view.  
  
    :param state: True to enable or False to disable default model  
    component items.  
    :type state: bool  
    """  
  
    self._item_model.enable_default_items(state)  
  
    # If false, deselect default items  
    if not state:  
        selection_model = self.lst_model_items.selectionModel()  
        selected_idxs = selection_model.selectedRows()  
        for sel_idx in selected_idxs:  
            item = self._item_model.item(sel_idx.row(), 0)  
            # If not enabled then deselect  
            if not item.isEnabled():  
                selection_model.select(sel_idx, QtCore.QItemSelectionModel.Deselect)
```

load

 **Code:**

```
load()
```

Subclass to determine how to initialize the items.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

130
131
132

 **Code:**

```
def load(self):
    """Subclass to determine how to initialize the items."""
    pass
```

model_names

 **Code:**

```
model_names()
```

Gets the names of the components in the item model.

Returns:

Type Description

list Returns the model names in lower case or an empty list if the item model has not been set.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

```
292
293      Code:
294
295
296
297
298
299     def model_names(self) -> typing.List[str]:
300         """Gets the names of the components in the item model.
301
302         :returns: Returns the model names in lower case or an empty
303             list if the item model has not been set.
304         :rtype: list
305
306         """
307
308         if self._item_model is None:
309             return []
310
311         model_components = self._item_model.model_components()
312
313         return [mc.name.lower() for mc in model_components]
```

selected_items

 **Code:**

```
selected_items()
```

Returns the selected items in the list view.

Returns:

Type Description

list A collection of the selected model component items. Returns an empty list if the item model has not been set.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

```
264
265      Code:
266
267
268
269
270
271     def selected_items(self) -> typing.List[ModelComponentItemType]:
272         """Returns the selected items in the list view.
273
274         :returns: A collection of the selected model component items. Returns
275         an empty list if the item model has not been set.
276         :rtype: list
277
278         """
279
280         if self._item_model is None:
281             return []
282
283         selection_model = self.lst_model_items.selectionModel()
284         idxs = selection_model.selectedRows()
285
286         return [self._item_model.item(idx.row()) for idx in idxs]
```

set_description

 **Code:**

```
set_description(description)
```

Updates the text for the selected item.

Parameters:

Name	Type	Description	Default
description	str	Description for the selected item.required	

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

```
186  
187  
188  
189  
190  
191  
192  
193
```

 **Code:**

```
def set_description(self, description: str):  
    """Updates the text for the selected item.  
  
    :param description: Description for the selected item.  
    :type description: str  
    """  
    self.txt_item_description.setText(description)  
    self.txt_item_description.setToolTip(description)
```

NcsComponentWidget

Bases: [ModelComponentWidget](#)

Widget for displaying and managing NCS pathways.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

```
342
343     Code:
344
345
346
347
348
349
def __init__(self, parent=None):
    super().__init__(parent)

    self.item_model = NcsPathwayItemModel(parent)

    self.lst_model_items.setDragEnabled(True)
    self.lst_model_items.setDragDropMode(QtWidgets.QAbstractItemView.DragOnly)
    self.lst_model_items.setAcceptDrops(False)
```

add_ncs_pathway

Code:

```
add_ncs_pathway(ncs_pathway)
```

Adds an NCS pathway object to the view.

Parameters:

Name	Type	Description	Default
ncs_pathway	NcsPathway	NCS pathway object to be added to the view.required	

Returns:

Type	Description
bool	Returns True if the NcsPathway was successfully added, else False.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

```
351
352
353
354
355
356
357
358     def add_ncs_pathway(self, ncs_pathway: NcsPathway) -> bool:
359         """Adds an NCS pathway object to the view.
360
361             :param ncs_pathway: NCS pathway object to be added to the view.
362             :type ncs_pathway: NcsPathway
363
364             :returns: Returns True if the NcsPathway was successfully added,
365             else False.
366             :rtype: bool
367             """
368
369         return self.item_model.add_ncs_pathway(ncs_pathway)
```

clear

Code:

```
clear()
```

Removes all NCS pathway items in the view.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

```
363  
364  
365  
366  
367
```

 **Code:**

```
def clear(self):  
    """Removes all NCS pathway items in the view."""  
    items = self.ncs_items()  
    for item in items:  
        self.item_model.remove_ncs_pathway(item.uuid)
```

load

 **Code:**

```
load()
```

Load items from settings.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

```
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482
```

 **Code:**

```
def load(self):  
    """Load items from settings."""  
    self.clear()  
  
    settings_manager.update_ncs_pathways()  
    ncs_pathways = settings_manager.get_all_ncs_pathways()  
  
    progress_dialog = QtWidgets.QProgressDialog(self)  
    progress_dialog.setWindowTitle(self.tr("Load NCS Pathways"))  
    progress_dialog.setMinimum(0)  
    progress_dialog.setMaximum(len(ncs_pathways))  
    progress_dialog.setLabelText(self.tr("Updating NCS pathways..."))  
    for i, ncs in enumerate(ncs_pathways, start=1):  
        progress_dialog.setValue(i)  
        if progress_dialog.wasCanceled():  
            break  
        self.add_ncs_pathway(ncs)
```

ncs_items **Code:**

```
ncs_items()
```

Returns a collection of all NcsPathwayItem objects in the list view.

Returns:**Type Description**

`list` Collection of NcsPathwayItem objects in the list view.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py`

381
382
383
384
385
386
387
388

 **Code:**

```
def ncs_items(self) -> typing.List[NcsPathwayItem]:  
    """Returns a collection of all NcsPathwayItem objects in the  
    list view.  
  
    :returns: Collection of NcsPathwayItem objects in the list view.  
    :rtype: list  
    """  
    return self.item_model.model_component_items()
```

pathways

 **Code:**

```
pathways(valid_only=False)
```

Returns a collection of NcsPathway objects in the list view.

Parameters:

Name	Type	Description	Default
<code>valid_only</code>	<code>bool</code>	True to only return those NcsPathway objects that are valid, default is <code>False</code> . <code>False</code> .	<code>False</code>

Returns:**Type Description**

`list` Collection of NcsPathway objects in the list view.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/model_component_widget.py

```
369
370      Code:
371
372
373
374
375
376     def pathways(self, valid_only=False) -> typing.List[NcsPathway]:
377         """Returns a collection of NcsPathway objects in the list view.
378
379             :param valid_only: True to only return those NcsPathway objects that
380             are valid, default is False.
381             :type valid_only: bool
382
383             :returns: Collection of NcsPathway objects in the list view.
384             :rtype: list
385         """
386
387         return self.item_model.pathways(valid_only)
```

NCS pathway editor

Dialog for creating or editing an NCS pathway entry.

NcsPathwayEditorDialog

 **Code:**

```
NcsPathwayEditorDialog(  
    parent=None, ncs_pathway=None, excluded_names=None  
)
```

Bases: `QDialog`, `WidgetUi`

Dialog for creating or editing an NCS pathway entry.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/ncs_pathway_editor_dialog.py`

30
31  **Code:**
32
33
34
35
36
37 def __init__(self, parent=None, ncs_pathway=None, excluded_names=None):
38 super().__init__(parent)
39 self.setupUi(self)
40
41
42
43 QgsGui.enableAutoGeometryRestore(self)
44
45
46 self._message_bar = QgsMessageBar()
47 self.vl_notification.addWidget(self._message_bar)
48
49
50
51 self._carbon_model = CarbonLayerModel(self)
52 self.lst_carbon_layers.setModel(self._carbon_model)
53 self.lst_carbon_layers.selectionModel().selectionChanged.connect(
54 self._on_selection_changed
55)
56
57
58
59
60 self.buttonBox.accepted.connect(self._on_accepted)
61 self.btn_add_layer.clicked.connect(self._on_select_file)
62
63
64
65 icon_pixmap = QtGui.QPixmap(ICON_PATH)
66 self.icon_la.setPixmap(icon_pixmap)
67
68
69
70 help_icon = FileUtils.get_icon("mActionHelpContents.svg")
71 self.btn_help.setIcon(help_icon)
72 self.btn_help.clicked.connect(self.open_help)
73
74
75
76 add_icon = FileUtils.get_icon("symbologyAdd.svg")
77 self.btn_add_carbon.setIcon(add_icon)
78 self.btn_add_carbon.clicked.connect(self._on_add_carbon_layer)
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179

```
self.btn_delete_carbon.setEnabled(False)
self.btn_delete_carbon.clicked.connect(self._on_remove_carbon_layer)

edit_icon = FileUtils.get_icon("mActionToggleEditing.svg")
self.btn_edit_carbon.setIcon(edit_icon)
self.btn_edit_carbon.setEnabled(False)
self.btn_edit_carbon.clicked.connect(self._on_edit_carbon_layer)

self._excluded_names = excluded_names
if excluded_names is None:
    self._excluded_names = []

self._edit_mode = False
self._layer = None
self._ncs_pathway = ncs_pathway
if self._ncs_pathway is not None:
    self._edit_mode = True
    self._layer = self._ncs_pathway.to_map_layer()
    self._update_controls()
```

edit_mode

 **Code:**

edit_mode

Returns the state of the editor.

Returns:

Type Description

bool True if the editor is editing an existing NcsPathway object, else False if its creating a new object.

`layer` `property`**Code:**`layer`

Returns the raster layer specified by the user, either existing layers in the map canvas or from the selected file.

Returns:

Type	Description
------	-------------

`QgsRasterLayer` The raster layer specified by the user or None if not set.

`ncs_pathway` `property`**Code:**`ncs_pathway`

Returns a reference to the NcsPathway object.

Returns:

Type	Description
------	-------------

`NcsPathway` Reference to the NcsPathway object.

`open_help`**Code:**`open_help(activated)`

Opens the user documentation for the plugin in a browser.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/ncs_pathway_editor_dialog.py

328
329
330

 **Code:**

```
def open_help(self, activated: bool):
    """Opens the user documentation for the plugin in a browser."""
    open_documentation(USER_DOCUMENTATION_SITE)
```

selected_carbon_items

 **Code:**

```
selected_carbon_items()
```

Returns the selected carbon items in the list view.

Returns:

Type Description

list A collection of the selected carbon items.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/ncs_pathway_editor_dialog.py

```
227  
228  
229  
230  
231  
232  
233  
234  
235  
236
```

 **Code:**

```
def selected_carbon_items(self) -> typing.List[CarbonLayerItem]:  
    """Returns the selected carbon items in the list view.  
  
    :returns: A collection of the selected carbon items.  
    :rtype: list  
    """  
  
    selection_model = self.lst_carbon_layers.selectionModel()  
    idxs = selection_model.selectedRows()  
  
    return [self._carbon_model.item(idx.row()) for idx in idxs]
```

validate

 **Code:**

```
validate()
```

Validates if name and layer have been specified.

Returns:

Type Description

True True if user input (i.e. name and layer) have been set.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/ncs_pathway_editor_dialog.py`

150
151  **Code:**
152
153
154
155
156
157 def validate(self) -> bool:
158 """Validates if name and layer have been specified.
159
160
161 :returns: True if user input (i.e. name and layer) have been set.
162 :rtype: True
163 """
164 status = True
165
166
167 self._message_bar.clearWidgets()
168
169
170 name = self.txt_name.text()
171
172 if not name:
173 msg = tr("NCS pathway name cannot be empty.")
174 self._show_warning_message(msg)
175 status = False
176
177
178 if name.lower() in self._excluded_names:
179 msg = tr("name has already been used.")
180 self._show_warning_message(f'''{name}' {msg}''')
181 status = False
182
183
184 if not self.txt_description.toPlainText():
185 msg = tr("Description cannot be empty.")
186 self._show_warning_message(msg)
187 status = False
188
189
190 layer = self._get_selected_map_layer()
191 if layer is None:
192 msg = tr("Map layer not specified.")
193 self._show_warning_message(msg)
194 status = False

```
if layer and not layer.isValid():
    msg = tr("Map layer is not valid.")
    self._show_warning_message(msg)
    status = False

return status
```

Style pixel value editor

Dialog for setting the pixel value for styling IMs.

PixelValueEditorDialog

 **Code:**

```
PixelValueEditorDialog(parent=None)
```

Bases: `QDialog`, `WidgetUi`

Dialog for setting the pixel value for styling IMs.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/pixel_value_editor_dialog.py`

27
28  **Code:**
29
30
31
32
33
34 def __init__(self, parent=None):
35 super().__init__(parent)
36 self.setupUi(self)
37
38
39
40 QgsGui.enableAutoGeometryRestore(self)
41
42
43
44 icon_pixmap = QtGui.QPixmap(ICON_PATH)
45 self.icon_la.setPixmap(icon_pixmap)
46
47
48
49 help_icon = FileUtils.get_icon("mActionHelpContents.svg")
50 self.btn_help.setIcon(help_icon)
51 self.btn_help.clicked.connect(self.open_help)
52
53
54
55 self._item_model = QtGui.QStandardItemModel(self)
 self._item_model.setColumnCount(1)
 self.tv_implementation_model.setModel(self._item_model)
 self.tv_implementation_model.setDragEnabled(True)
 self.tv_implementation_model.setAcceptDrops(True)
 self.tv_implementation_model.setShowGrid(False)
 self.tv_implementation_model.setDragDropOverwriteMode(False)
 self.tv_implementation_model.setDragDropMode(
 QtWidgets.QAbstractItemView.InternalMove
)
 self.tv_implementation_model.horizontalHeader().setSectionResizeMode(
 QtWidgets.QHeaderView.Stretch
)
 self._load_items()

`item_mapping` `property` **Code:**`item_mapping`

Returns a mapping of the implementation model position in the table and its corresponding unique identifier.

We are using an OrderedDict to ensure consistency across different Python versions in the different platforms that QGIS runs on.

Returns:

Type	Description
------	-------------

<code>OrderedDict</code>	The mapping of the implementation model position in the table and its corresponding unique identifier.
--------------------------	--

[open_help](#) **Code:**`open_help(activated)`

Opens the user documentation for the plugin in a browser.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/pixel_value_editor_dialog.py`

70
71
72 **Code:**

```
def open_help(self, activated: bool):
    """Opens the user documentation for the plugin in a browser."""
    open_documentation(USER_DOCUMENTATION_SITE)
```

Priority weighting layer dialog

Priority layer dialog

PriorityLayerDialog

 **Code:**

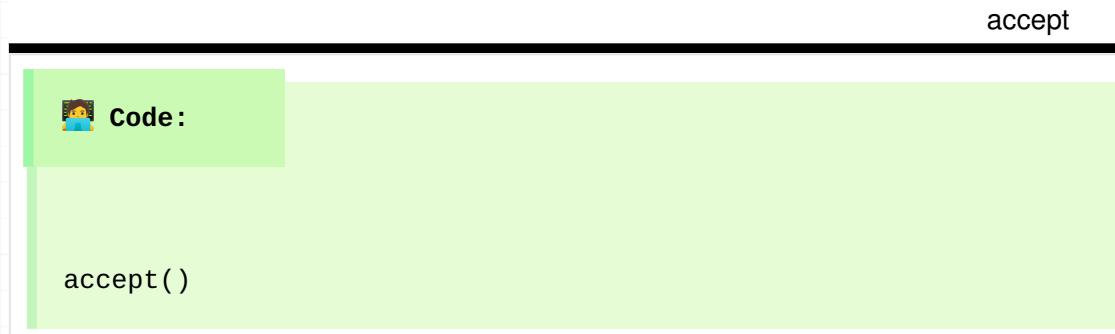
```
PriorityLayerDialog(layer=None, parent=None)
```

Bases: `QDialog`, `DialogUi`

Dialog that provide UI for priority layer details.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/priority_layer_dialog.py`

35
36  **Code:**
37
38
39
40
41
42 def __init__(
43 self,
44 layer=None,
45 parent=None,
46):
47 super().__init__(parent)
48 self.setupUi(self)
49 self.layer = layer
50
51
52 self.button_box.button(QtWidgets.QDialogButtonBox.Ok).setEnabled(False)
53
54
55 self.map_layer_box.layerChanged.connect(self.map_layer_changed)
56
57
58 ok_signals = [
59 self.layer_name.textChanged,
60 self.layer_description.textChanged,
61 self.map_layer_file_widget.fileChanged,
62 self.map_layer_box.layerChanged,
63]
64
65
66 for signal in ok_signals:
67 signal.connect(self.update_ok_buttons)
68
69 icon_pixmap = QtGui.QPixmap(ICON_PATH)
70 self.icon_la.setPixmap(icon_pixmap)
71
72 self._user_defined = True
73
74 self.models = []
75 self.initialize_ui()



Handles logic for adding new priority layer and edit existing one

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/priority_layer_dialog.py

182
183  **Code:**
184
185
186
187
188
189
190 def accept(self):
191 """Handles logic for adding new priority layer and edit existing one"""
192 layer_id = uuid.uuid4()
193 layer_groups = []
194 layer = {}
195 if self.layer is not None:
196 layer_id = self.layer.get("uuid")
197 layer_groups = self.layer.get("groups", [])
198
199 layer["uuid"] = str(layer_id)
200 layer["name"] = self.layer_name.text()
201 layer["description"] = self.layer_description.toPlainText()
202 layer["groups"] = layer_groups
203
204
205 layer["path"] = self.map_layer_file_widget.filePath()
206 layer[USER_DEFINED_ATTRIBUTE] = self._user_defined
207
208 settings_manager.save_priority_layer(layer)
209
210 self.layer = layer
211 self.set_selected_items(self.models)
212
213 super().accept()

initialize_ui **Code:**`initialize_ui()`

Populate UI inputs when loading the dialog

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/priority_layer_dialog.py`

Code :

```
        if str(self.layer.get("uuid")) in model_layer_uuids:  
            self.models.append(model)  
  
        self.set_selected_items(self.models)  
  
        self._user_defined = self.layer.get(USER_DEFINED_ATTRIBUTE, True)
```

map_layer_changed

Code:

```
map_layer_changed(layer)
```

Sets the file path of the selected layer in file path input

Parameters:

Name	Type	Description	Default
layer	QgsMapLayer	Qgis map layer	required

• Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/priority_layer_dialog.py

Code:

```
66  
67  
68  
69  
70  
71  
72  
73 def map_layer_changed(self, layer):  
    """Sets the file path of the selected layer in file path input  
  
    :param layer: Qgis map layer  
    :type layer: QgsMapLayer  
    """  
    if layer is not None:  
        self.map_layer_file_widget.setFilePath(layer.source())
```

open_help **Code:**`open_help()`

Opens the user documentation for the plugin in a browser

- Source code in /home/samwelij/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/priority_layer_dialog.py

206
207
208

 **Code:**

```
def open_help(self):
    """Opens the user documentation for the plugin in a browser"""
    open_documentation(USER_DOCUMENTATION_SITE)
```

open_layer_select_dialog **Code:**`open_layer_select_dialog()`

Opens priority layer item selection dialog

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/priority_layer_dialog.py

130
131
132
133 **Code:**

```
def open_layer_select_dialog(self):
    """Opens priority layer item selection dialog"""
    model_select_dialog = ItemsSelectionDialog(self, self.layer,
    model_select_dialog.exec_()
```

set_selected_items

 **Code:**

```
set_selected_items(models, removed_models=[])
```

Adds this dialog layer into the passed models and removes it from the unselected models passed as removed_models.

Parameters:

Name	Type	Description	Default
models	list	Selected implementation models	required
removed_models	list	Implementation models that dialog layer should be removed from.	[]

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/priority_layer_dialog.py`

135
136
137  **Code:**
138
139
140
141
142 def set_selected_items(self, models, removed_models=[]):
143 """Adds this dialog layer into the passed models and removes it from the
144 unselected models passed as removed_models.
145
146 :param models: Selected implementation models
147 :type models: list
148
149 :param removed_models: Implementation models that dialog
150 layer should be removed from.
151 :type removed_models: list
152
153 """
154
155
156
157
158
159
160
161
162 self.models = models
163
164
165 models_names = [model.name for model in models]
166 self.selected_models_le.setText(" , ".join(models_names))
167
168
169
170 if not self.layer:
171 return
172
173
174
175 if len(removed_models) <= 0:
176 all_models = settings_manager.get_all_implementation_models()
177 removed_models = [
178 model for model in all_models if model.name not in models_names
179]
180
181
182 for model in models:
183 models_layer_uuids = [
184 str(layer.get("uuid"))
185 for layer in model.priority_layers

```
        if layer is not None
    ]
    if (
        self.layer is not None
        and str(self.layer.get("uuid")) not in models_layer_uuids
    ):
        model.priority_layers.append(self.layer)
        settings_manager.save_implementation_model(model)
    for model in removed_models:
        for layer in model.priority_layers:
            if layer is None:
                continue
            if str(layer.get("uuid")) == str(self.layer.get("uuid")):
                model.priority_layers.remove(layer)
                settings_manager.save_implementation_model(model)
```

update_ok_buttons

 **Code:**

update_ok_buttons()

Responsible for changing the state of the dialog OK button.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/priority_layer_dialog.py

 **Code:**

```
75
76
77
78
79
80
81
82 def update_ok_buttons(self):
83     """Responsible for changing the state of the
84     dialog OK button.
85     """
86
87     enabled_state = (
88         self.layer_name.text() != ""
89         and self.layer_description.toPlainText() != ""
90         and (
91             self.map_layer_box.currentLayer() is not None
92             or (
93                 self.map_layer_file_widget.filePath() is not None
94                 and self.map_layer_file_widget.filePath() is not ""
95             )
96         )
97     )
98
99     self.button_box.button(QtWidgets.QDialogButtonBox.Ok).setEnabled(enabled_stan
```

Priority group

Priority group item widget

PriorityGroupWidget



```
PriorityGroupWidget(group, parent=None)
```

Bases: QWidget , WidgetUi

Widget that provide UI for priority group details.

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/priority_group_widget.py`

Code

```
36 def __init__(  
37     self,  
38     group,  
     parent=None,  
):  
    super().__init__(parent)  
    self.setupUi(self)  
    self.group = group  
  
    self.initializeUi()
```

group_value**Code:**`group_value()`

Returns: name (dict): Priority group value

- Source code in `/home/samwel1/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/priority_group_widget.py`

```
72  
73  
74  
75  
76  
77
```

```
def group_value(self):  
    """  
    Returns:  
        name (dict): Priority group value  
    """  
    return self.group_slider.value()
```

initialize_ui**Code:**`initialize_ui()`

Populate UI inputs when loading the widget

- Source code in `/home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/priority_group_widget.py`

```
40
41  Code:
42
43
44
45
46
47
48
49

def initialize_ui(self):
    """Populate UI inputs when loading the widget"""

    if self.group is not None:
        self.group_la.setText(self.group.get("name"))
        self.group_slider.setValue(int(self.group.get("value", 0)))
        self.group_spin_box.setValue(int(self.group.get("value", 0)))

    self.group_slider.valueChanged.connect(self.update_spin_box)
    self.group_spin_box.valueChanged.connect(self.update_slider)
```

name

Code :

name()

Returns: name (dict): Priority group name

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/priority_group_widget.py

```
65
66  Code:
67
68
69
70

def name(self):
    """
    Returns:
        name (dict): Priority group name
    """
    return self.group.get("name")
```

set_group

 **Code:**

```
set_group(group)
```

Sets the priority layer group and updates the slider and input values

Args: group (dict): Priority group

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/priority_group_widget.py

```
51
52      Code:
53
54
55
56
57
58     def set_group(self, group):
59         """Sets the priority layer group and updates the slider and
60         input values
61
62     Args:
63         group (dict): Priority group
64     """
65
66
67     if group is not None:
68         self.group = group
69         self.group_la.setText(group.get("name"))
70         self.group_slider.setValue(int(group["value"]))
71         self.group_spin_box.setValue(int(group["value"]))
```

update_slider

 **Code:**

```
update_slider(value)
```

Changes the current slider value.

Args: value (int): Value to be set on the slider Note: Emits input_value_changed signal

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/priority_group_widget.py

```
79
80  Code:
81
82
83
84
85
86 def update_slider(self, value):
87     """Changes the current slider value.
88
89     Args:
90         value (int): Value to be set on the slider
91
92     Note:
93         Emits input_value_changed signal
94
95     """
96
97     if self.group is not None:
98         self.input_value_changed.emit(self.group["name"], value)
99         self.group_slider.blockSignals(True)
100        self.group_slider.setValue(value)
101        self.group_slider.blockSignals(False)
```

update_spin_box

 **Code:**

```
update_spin_box(value)
```

Changes the input value of the spin box

Args: value (int): Value to be set on the spin box. Note: Emits slider_value_changed signal

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/priority_group_widget.py

```
93
94
95  Code:
96
97
98
99
100
101
102
103
104
105
106
def update_spin_box(self, value):
    """Changes the input value of the spin box

    Args:
        value (int): Value to be set on the spin box.

    Note:
        Emits slider_value_changed signal

    """
    if self.group is not None:
        self.slider_value_changed.emit(self.group["name"], value)
        self.group_spin_box.blockSignals(True)
        self.group_spin_box.setValue(value)
        self.group_spin_box.blockSignals(False)
```

widgets

```
 Code:
 widgets()
```

Returns widget_list (list): List of component widgets for the priority group widget

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/priority_group_widget.py

108
109  **Code:**

110
111
112
113
114

```
def widgets(self) -> typing.List[QtWidgets.QWidget]:  
    """  
    Returns  
        widget_list (list): List of component  
        widgets for the priority group widget  
    """  
    return [self.group_la, self.group_slider, self.group_spin_box]
```

Progress dialog

Analysis progress dialog file

ProgressDialog

 **Code:**

```
ProgressDialog(  
    message=None,  
    minimum=0,  
    maximum=100,  
    main_widget=None,  
    parent=None,  
    scenario_id=None,  
    scenario_name=None,  
)
```

Bases: `QDialog`, `Ui_DlgProgress`

Progress dialog class

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/progress_dialog.py

33
34  **Code:**

35
36
37
38
39
40 def __init__(
41 self,
42 message=None,
43 minimum=0,
44 maximum=100,
45 main_widget=None,
46 parent=None,
47 scenario_id=None,
48 scenario_name=None,
49):
50 super().__init__(parent)
51 self.setupUi(self)
52 self.scenario_id = scenario_id
53 self.scenario_name = scenario_name
54
55 self.main_widget = main_widget
56 self.report_manager = report_manager
57
58 self.analysis_task = None
59
60 # Dialog window flags
61 flags = QtCore.Qt.WindowMinimizeButtonHint | QtCore.Qt.WindowCloseButtonHint
62 self.setWindowFlags(flags)
63
64 # Dialog statuses
65 self.analysis_running = True
66
67 if message is None:
68 self.change_status_message(tr("Starting processing"))
69 else:
70 self.change_status_message(message)

```
90
91     if scenario_name:
92         self.title.setText(
93             f"{self.title.text()} for scenario <b>{self.scenario_name}</b>"
94         )
95
96
97
98     # Report status
99     self.report_running = False
100
101
102
103     # Progress bar
104     self.progress_bar.setAlignment(QtCore.Qt.AlignLeft | QtCore.Qt.AlignVCenter)
105     self.progress_bar.setMinimum(minimum)
106     self.progress_bar.setMaximum(maximum)

# Report menu
self.menu = QMenu("&View Report")
self.btn_view_report.setMenu(self.menu)
self.btn_view_report.setIcon(QIcon(ICON_REPORT))

# Menu button to open report in Layout designer
self.designer_action = QAction(
    QIcon(ICON_LAYOUT), "Layout designer", parent=self
)
self.designer_action.triggered.connect(self.view_report_layout_designer)
self.designer_action.setEnabled(False)
self.menu.addAction(self.designer_action)

# Open a PDF version of the report
self.pdf_action = QAction(QIcon(ICON_PDF), "Open PDF", parent=self)
self.pdf_action.triggered.connect(self.view_report_pdf)
self.pdf_action.setEnabled(False)
self.menu.addAction(self.pdf_action)

# Open a Help for reports
action = QAction(QIcon(ICON_HELP), "Help", parent=self)
action.triggered.connect(self.open_report_help)
```

```
action.setEnabled(True)
self.menu.addAction(action)

# Connections
self.btn_cancel.clicked.connect(self.cancel_clicked)

self.analysis_finished_message = tr("Analysis has finished.")
```

cancel_clicked

 **Code:**

```
cancel_clicked()
```

User clicked cancel.

Processing will be stopped, and the UI will be updated to accommodate the processing status.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/progress_dialog.py

```
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215
```

 **Code:**

```
def cancel_clicked(self) -> None:  
    """User clicked cancel.  
  
    Processing will be stopped, and the UI will be updated to accommodate  
    the processing status.  
    """  
  
    self.analysis_cancelled.emit()  
  
    self.cancel_reporting()  
  
    if self.analysis_running:  
        # If cancelled is clicked  
        self.stop_processing()  
        if self.analysis_task:  
            self.analysis_task.processing_cancelled = True  
            self.analysis_task.cancel()  
    else:  
        # If close has been clicked. In this case processing were already stopped  
        super().close()
```

cancel_reporting

 **Code:**

```
cancel_reporting()
```

Cancel the report generation process.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/progress_dialog.py

217
218
219
220

 **Code:**

```
def cancel_reporting(self):
    """Cancel the report generation process."""
    if self.report_running:
        self.report_manager.remove_report_task(self.scenario_id)
```

change_status_message

 **Code:**

```
change_status_message(message=None)
```

Updates the status message

Parameters:

Name	Type	Description	Default
message	str	Message to show on the status bar	None

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/progress_dialog.py

```
146
147     Code:
148
149
150
151
152
153     def change_status_message(self, message=None) -> None:
154         """Updates the status message
155
156         :param message: Message to show on the status bar
157         :type message: str
158         """
159
160
161         if message:
162             self.lbl_status.setText(message)
```

get_processing_status

Code:

```
get_processing_status()
```

Returns the status of the processing.

Returns:

Type Description

bool Status of processing.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/progress_dialog.py

```
116  
117      Code:  
118  
119  
120  
121  
122  
123  
  
def get_processing_status(self) -> bool:  
    """Returns the status of the processing.  
  
    :returns: Status of processing.  
    :rtype: bool  
    """  
  
    return self.analysis_running
```

get_progress_bar

 **Code:**

```
get_progress_bar()
```

Returns a reference to the Progress bar object.

Returns:

Type	Description
QProgressBar	Progress bar

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/progress_dialog.py

```
125
126  Code:
127
128
129
130
131
132
def get_progress_bar(self) -> QProgressBar:
    """Returns a reference to the Progress bar object.

    :returns: Progress bar
    :rtype: QProgressBar
    """
    return self.progress_bar
```

open_report_help

Code :

open_report_help()

[Opens the Report guide in a browser](#)

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/progress_dialog.py

193

194

195

```
def open_report_help(self) -> None:  
    """Opens the Report guide in a browser"""  
    open_documentation(REPORT_DOCUMENTATION)
```

processing_cancelled **Code:**

```
processing_cancelled()
```

Post-steps when processing were cancelled.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/progress_dialog.py

```
248  
249  
250  
251  
252  
253  
254  
255
```

 **Code:**

```
def processing_cancelled(self) -> None:  
    """Post-steps when processing were cancelled."""  
  
    self.analysis_running = False  
  
    # Change cancel button to the close button status  
    self.btn_cancel.setText(tr("Close"))  
    self.btn_view_report.setEnabled(False)
```

processing_finished **Code:**

```
processing_finished()
```

Post-steps when processing succeeded.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/progress_dialog.py

```
257
258      Code:
259
260
261
262
263
264     def processing_finished(self) -> None:
265         """Post-steps when processing succeeded."""
266
267
268         self.analysis_running = False
269         self.change_status_message(self.analysis_finished_message)
270
271         # Change cancel button to the close button status
272         self.btn_cancel.setText(tr("Close"))
273         self.btn_view_report.setEnabled(True)
274         icon = self.style().standardIcon(QStyle.SP_DialogCloseButton)
275         self.btn_cancel.setIcon(icon)
```

reject

 **Code:**

```
reject()
```

Called when the dialog is closed

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/progress_dialog.py

```
222
223      Code:
224
225
226
227
228
229     def reject(self) -> None:
230         """Called when the dialog is closed"""
231         self.analysis_cancelled.emit()
232
233         if self.analysis_running:
234             # Stops analysis if it is still running
235             self.stop_processing()
236             if self.analysis_task:
237                 self.analysis_task.processing_cancelled = True
238                 self.analysis_task.cancel()
239
240             self.cancel_reporting()
241
242         super().reject()
```

run_dialog

 **Code:**

```
run_dialog()
```

Runs/opens the dialog. Sets modal to modeless. This will allow the dialog to display and not interfere with other processes.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/progress_dialog.py

108
109
110
111
112
113
114

 **Code:**

```
def run_dialog(self):
    """Runs/opens the dialog. Sets modal to modeless.
    This will allow the dialog to display and not interfere with other processes

    """
    self.setModal(False)
    self.show()
```

set_report_complete

 **Code:**

```
set_report_complete()
```

Enable layout designer and PDF report buttons.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/progress_dialog.py

```
156
157      Code:
158
159
160
161
162
163
def set_report_complete(self):
    """Enable layout designer and PDF report buttons."""
    self.btn_view_report.setEnabled(True)
    self.designer_action.setEnabled(True)
    self.pdf_action.setEnabled(True)
    self.report_running = False

    self.processing_finished()
```

stop_processing

```
 Code:
stop_processing()
```

The user cancelled the processing.

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/progress_dialog.py

237
238
239
240
241
242
243
244
245
246

 **Code:**

```
def stop_processing(self) -> None:  
    """The user cancelled the processing."""  
  
    self.change_status_message(tr("Processing has been cancelled by the user"))  
  
    # Stops the processing task  
    if self.main_widget:  
        self.main_widget.cancel_processing_task()  
  
    self.processing_cancelled()
```

update_progress_bar **Code:**

```
update_progress_bar(value)
```

Sets the value of the progress bar

Parameters:

Name	Type	Description	Default
value	float	Value to be set on the progress bar required	

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/progress_dialog.py

```
134
135     Code:
136
137
138
139
140
141     def update_progress_bar(self, value) -> None:
142         """Sets the value of the progress bar
143
144             :param value: Value to be set on the progress bar
145             :type value: float
146             """
147
148         if self.progress_bar:
149             try:
150                 self.progress_bar.setValue(int(value))
151             except RuntimeError:
152                 log(tr("Error setting value to a progress bar"), notifi...
```

view_report_layout_designer

Code:

```
view_report_layout_designer()
```

Opens the report in layout designer

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/progress_dialog.py

```
179
180      Code:
181
182
183
184
185
186     def view_report_layout_designer(self) -> None:
187         """Opens the report in layout designer"""
188         if not self.scenario_id:
189             log("Scenario ID has not been set.")
190             return
191
192
193         result = self.report_manager.report_result(self.scenario_id)
194         if result is None:
195             log("Report result not found.")
196         else:
197             status = self.report_manager.open_layout_designer(result)
198             if not status:
199                 log("Unable to open layout designer.")
```

view_report_pdf

 **Code:**

```
view_report_pdf()
```

Opens a PDF version of the report

- Source code in /home/samweli/Workspace/Projects/cplus-plugin/src/cplus_plugin/gui/progress_dialog.py

```
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177
```

 **Code:**

```
def view_report_pdf(self) -> None:  
    """Opens a PDF version of the report"""  
    if not self.scenario_id:  
        log("Scenario ID has not been set.")  
        return  
  
    result = self.report_manager.report_result(self.scenario_id)  
    if result is None:  
        log("Report result not found.")  
    else:  
        status = self.report_manager.view_pdf(result)  
        if not status:  
            log("Unable to open PDF report.")
```

1 About

1.1 Conservation International



{style="display: block; margin: 0 auto;

border: 1px solid grey" }

Through the help of community leaders and policymakers, the mission of [Conservation International \(CI\)](#) is to combat climate change and preserve carbon stores. CI and the Climate Positive Land Use Strategy (CPLUS) project need the ability to spatially analyze a given area to determine the best land use in order to naturally combat climate change and the effects thereof on a global scale. Further, reports need to be generated to communicate to stakeholders the importance of conserving and maintaining certain areas and the need to restore lost climate-positive environments. Thus, mitigating the negative effects of climate change produced by our daily activities. The aim is to ensure a better future for everyone.

1.2 Kartoza



{style="display: block; margin: 0 auto; border: 1px solid grey; width: 50%; height: 50%"}
[Kartoza](#)

Kartoza will make geospatial data and technology work for you by partnering with us for training, development and maintenance of GIS systems.

Kartoza is a South Africa-based Free and Open Source GIS (FOSSGIS) service provider. We use GIS software to solve complex location-related problems for individuals, businesses and governments around the world.

Our team develops software using FOSSGIS to give you the freedom to share and modify your GIS as your needs grow and change.

1.3 License



<https://github.com/kartoza/cplus-plugin/>