AUTOMATISK

GENERALISERING

Siden kartet har en mindre målestokk enn fenomenene det skal illustrere, må informasjonen forenkles og avgrenses. Dette innebærer at detaljer kan gå tapt, og bare de mest relevante elementene blir inkludert for å sikre at kartet forblir lett forståelig.

Kartverket

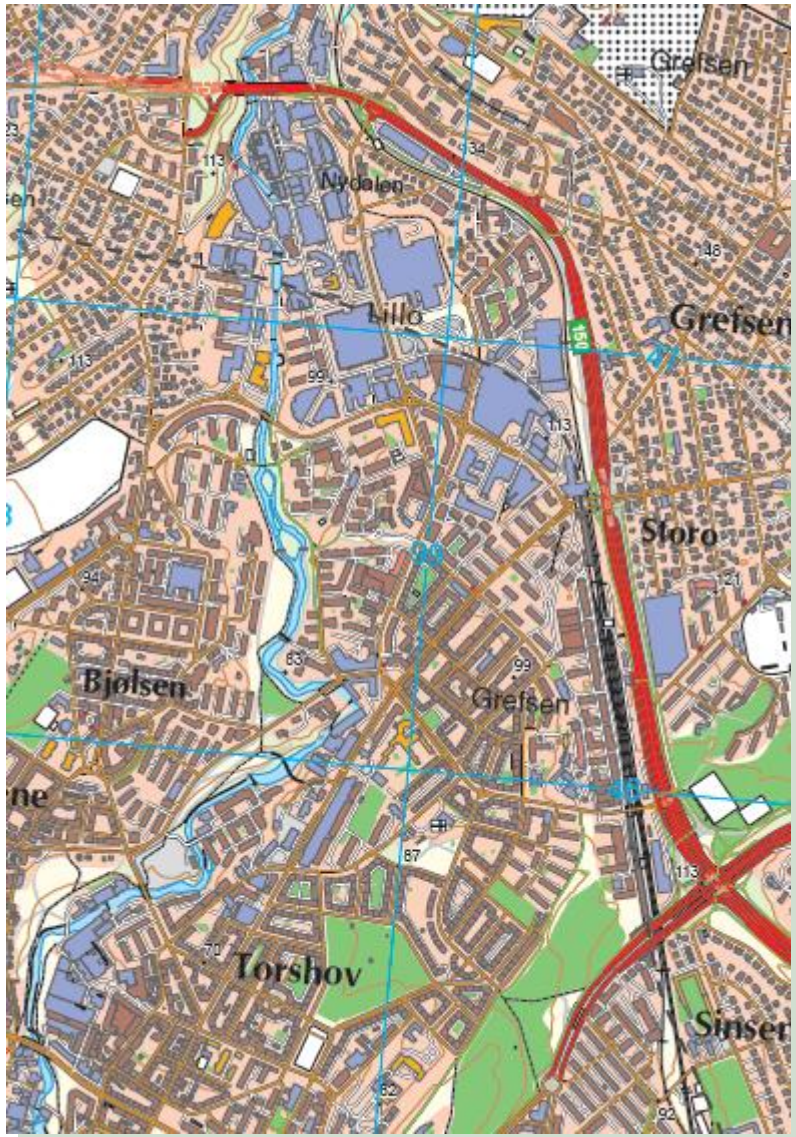
AUTOMATISK

GENERALISERING

Går fra en subjektiv og nøyaktig generaliseringstilnærming til en objektiv og mindre nøyaktig generalisering. Fra menneske til maskin….

Kartverket

GENERALISERING

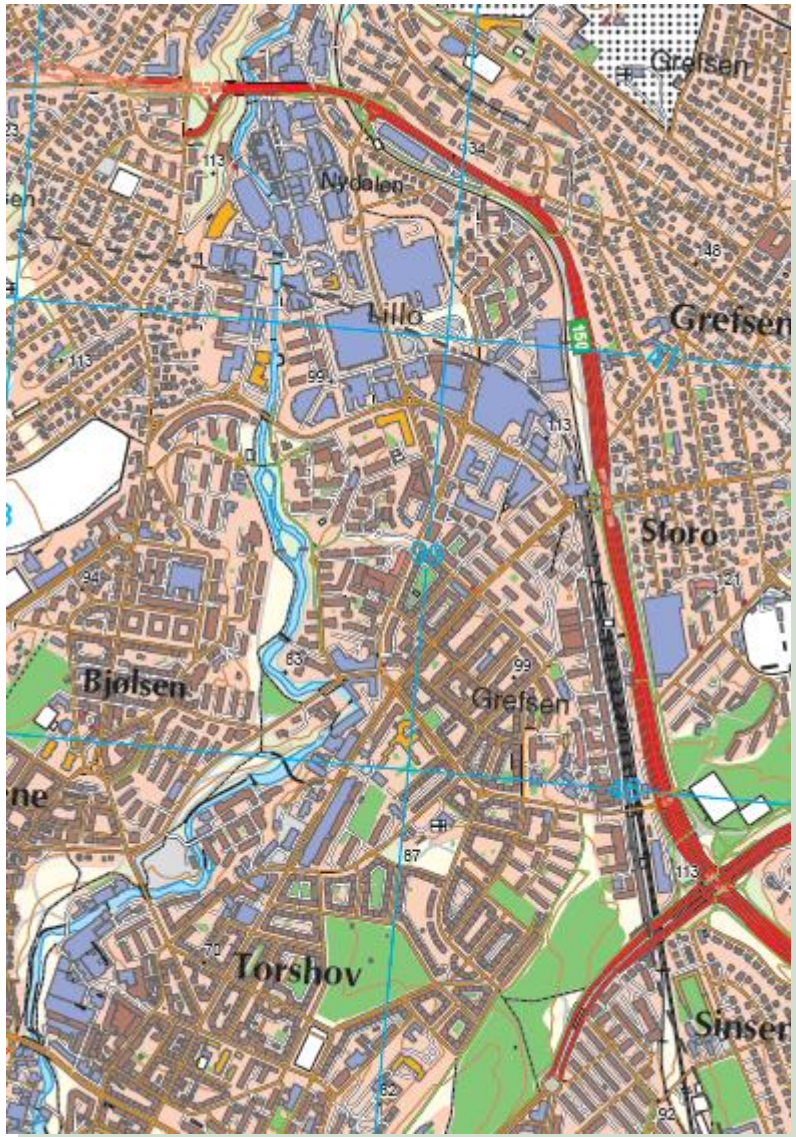


1:20 000

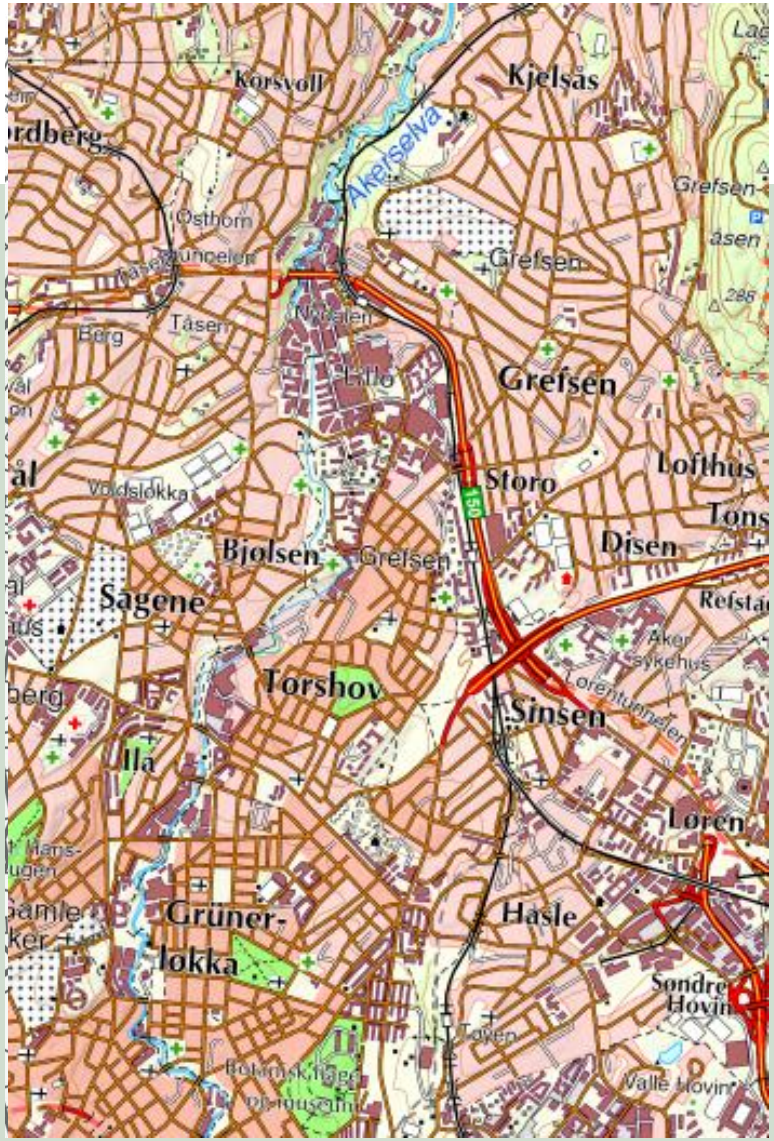
Samme innhold, zoomet ut

Kartverket

GENERALISERING



1:20 000



1:50 000

Kartverket

AG - TEAMET

Ida Hope Barth

Har jobbet 9 år i Kartverket

Strukturgeolog fra UiO

Elling Aronsen Oftedal

Har jobbet 1 år i Kartverket

Samfunnsgeografi UiO

Virginia Antonijevic

Har jobbet 14 år i Kartverket

Miljøvitenskap & GIS Argentina og UiB

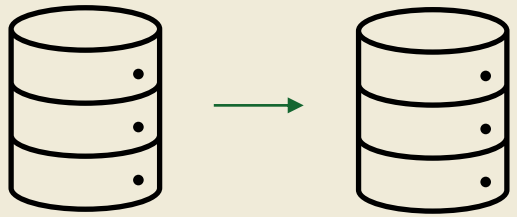
Mari Buseth Helland

Har jobbet 1 år i Kartverket

Naturgeografi, NTNU + Informatikk UiO

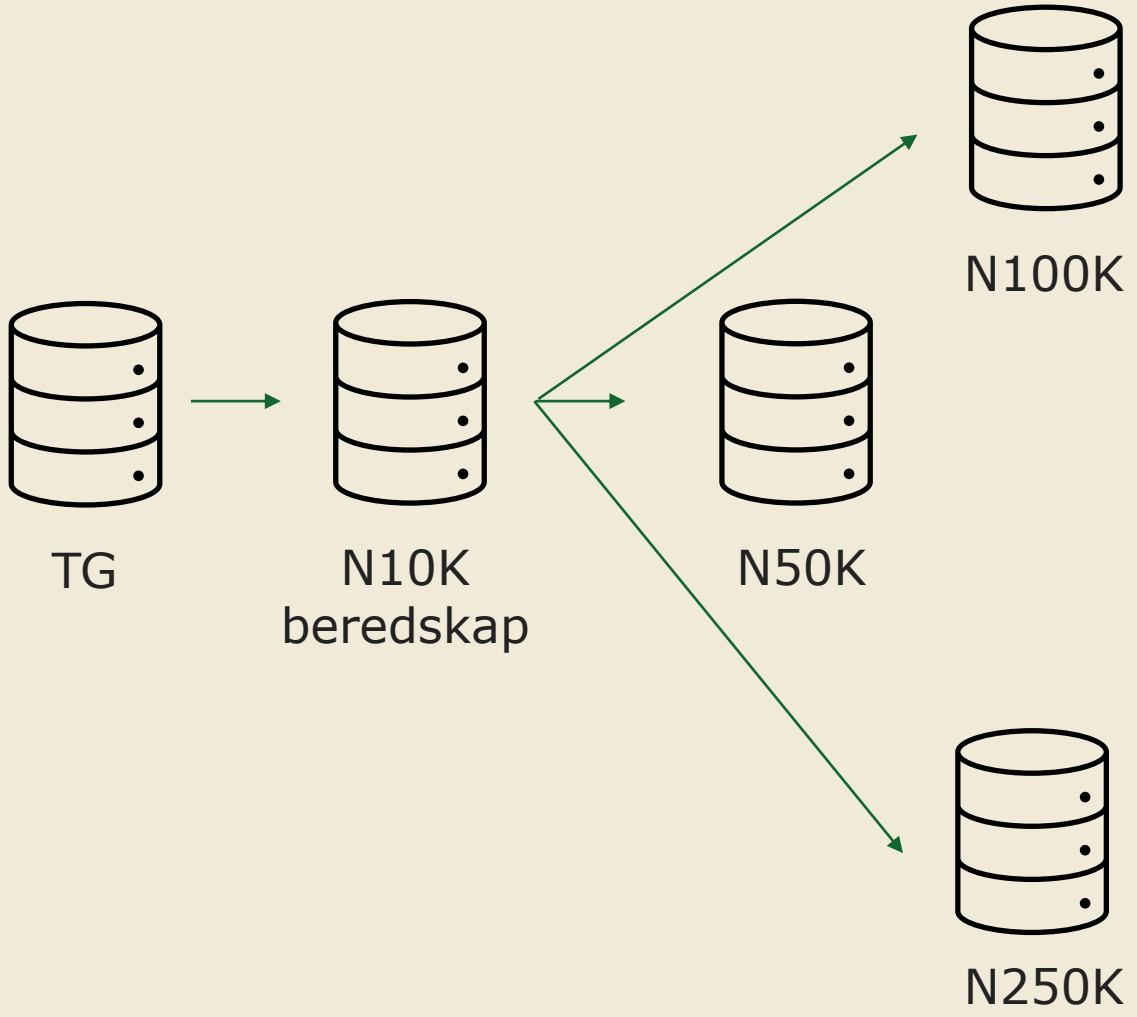
AG = Automatisk generalisering

NÅ



- Automatisk generalisering fra N50K til N100K
- Ukentlige oppdateringer, med unntak for bygninger, arealdekke, veier og elver.
- Stedsnavn kopierer vi fra N250K

PLANEN FREMOVER



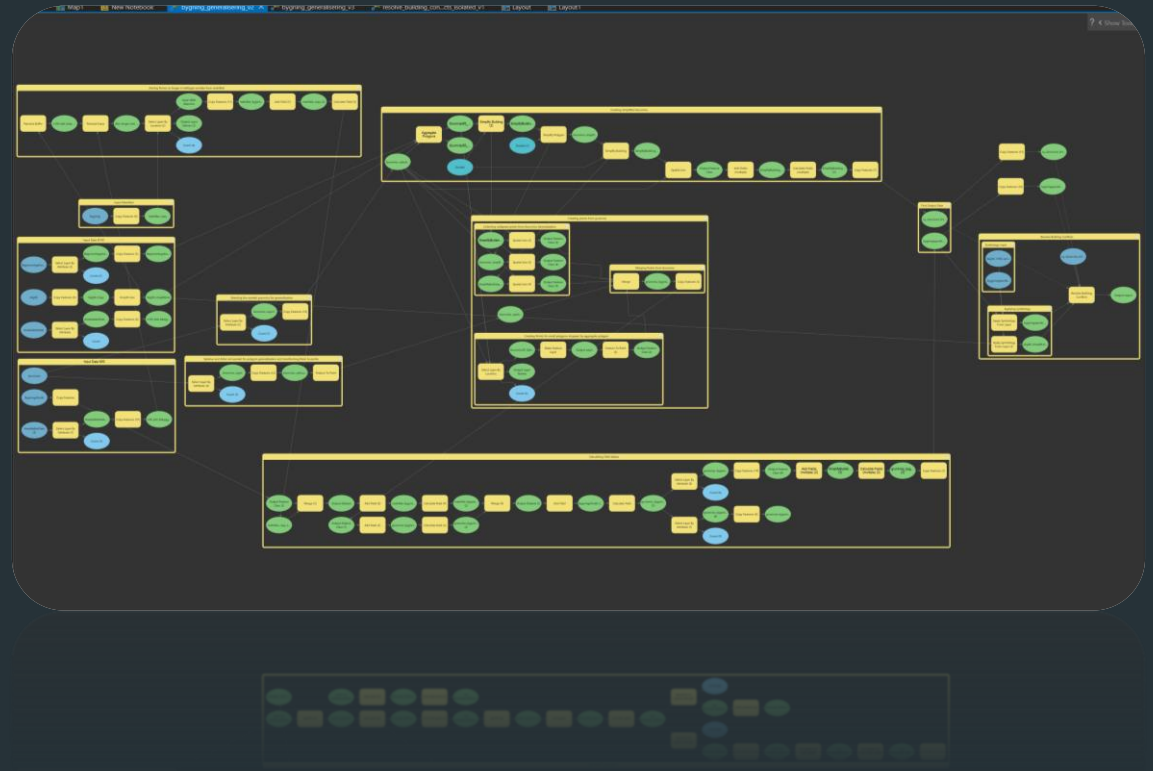
- Ny masterdatabase TG, som samler kartdata og sammenstiller. Ingen Kartografiske tilpasninger.
- Ut i fra TG skal vi lage en kartdatabase til det norske forsvaret.
- Grunnlaget for alle småskaladatabasene.

Oppdatere en gang og deretter generalisere.

Friction working with Model-Builder

- Fast to build, hard to maintain.
  - Hard to cooperate on the same project.
  - Lost work as team members have left project.
  - Harder to maintain with ArcGIS Pro updates (the input params can be lost).

- Hard to manage and keep documentation in the model.
  - It’s hard to keep a clean model and document “in model” comments.
  - Documentation separated from logic is hard to maintain and use.



Kartverket

Transition to ArcPy/Python

- Started from scratch with our building model which needed to be re-done.
  - Our initial focus was on research, learning and experimentation over immediate results.
  - Our goal is to reduce future tech debt and maintenance cost over initial speed.
  - Require understanding from organization that results will initially take longer.

- Can work with a team with varied experience in programming
  - Possible to split team with different tasks.
  - Requirement is that team members are interested in learning.

    → … and for organizations to invest in employees.

Kartverket

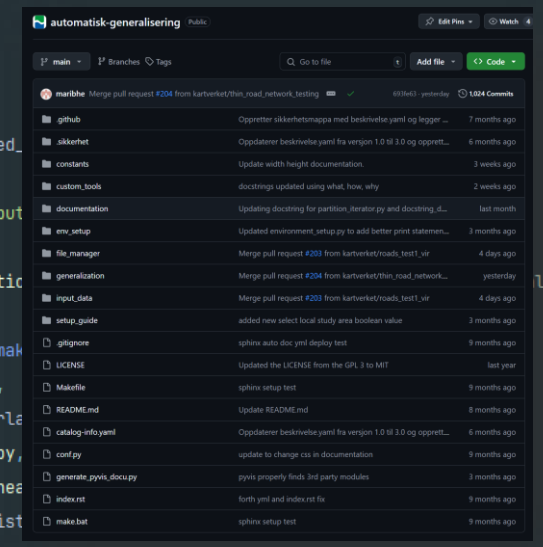
```
def prepare_input_data(self):  1 usage  EllingOftedalKV
    for alias, types in self.nested_alias_type_data.items():
        if "context" in types:
            context_data_path = types["context"]
            context_data_copy = (
                f"{self.root_file_partition_iterator}_{alias}_context_data_copy"
            )
            if self.selection_of_context_features:
                PartitionIterator.create_feature_class(
                    full_feature_path=context_data_copy,
                    template_feature=context_data_path,
                )

                for input_alias, input_types in self.nested_
                    if "input_copy" in input_types:
                        input_data_copy = input_types["input

                        context_features_near_input_selectio

                        custom_arcpy.select_location_and_mak
                            input_layer=context_data_path,
                            overlap_type=custom_arcpy.Overla
                            select_features=input_data_copy,
                            output_name=context_features_nea
                            search_distance=self.search_dist
                        )
                        arcpy.management
                            inputs=conte
                            target=conte
                            schema_type=
                        )
                        arcpy.management
                            context_feat
                        )
                print(f"Processed contex

            else:
                arcpy.management.Copy(
                    in_data=context_data
                    out_data=context_dat
                )
                print(f"Copied context data for: {alias}")
```



Scale and maintainability

- Object Oriented Input and Output paths
  - We use an Enum Class where the variables holds a path constructor.

    → Variable name == file name

```
data_preparation___road_symbology_buffers___n100_building = (
    file_manager.generate_file_name_gdb(
        script_source_name=data_preparation,
        description="road_symbology_buffers",
    )
)
```

```
road_lines_to_buffer_symbology = LineToBufferSymbology(
    input_road_lines=Building_N100.data_preparation___unsplit_roads___n100_building.value,
    sql_selection_query=N100_SQLResources.road_symbology_size_sql_selection.value,
    output_road_buffer=Building_N100.data_preparation___road_symbology_buffers___n100_building.value,
    write_work_files_to_memory=False,
    keep_work_files=False,
    root_file=Building_N100.data_preparation___root_file_line_symbology___n100_building.value,
    fixed_buffer_addition=N100_Values.rbc_barrier_clearance_distance_m.value,
)
road_lines_to_buffer_symbology.run()
```

```
def generate_file_name_gdb(  262 usages  EllingOftedalKV
    self,
    script_source_name: str,
    description: str,
):
    """
    Generates a file path for geodatabase (.gdb) files. After validating the input.

    Args:
        script_source_name (str): The name of the script or source generating the file.
        description (str): A brief description of the file's purpose or contents.

    Returns:
        str: The absolute path for the .gdb file.
    """
    self.validate_inputs( *args: script_source_name, description)
    return rf"{self.relative_path_gdb}\{script_source_name}___{description}___{self.scale}_{self.object}"
```

```
data_preparation___railway_station_points_from_n100___n100_building
data_preparation___railway_stations_to_polygons___n100_building
data_preparation___road_symbology_buffers___n100_building
data_preparation___unsplit_roads___n100_building
data_preparation___urban_area_selection_n100___n100_building
data_preparation___urban_area_selection_n100_buffer___n100_building
data_selection___begrensningskurve_n100_input_data___n100_building
```

Kartverket

Scale and maintainability

- Updated in code documentation
  - Project stability with new team members.
    - → And security if team members leave.
  - Keeping it updated catches errors.
- Generated using Sphinx
  - Have had good use with cooperation with Belgium.

```python
@timing_decorator  1 usage  maribhe
def reducing_clusters():
    """
    What:
        Reduces hospital and church clusters by keeping only one point for each detected cluster. This ensures that
        spatial redundancy is minimized, and each cluster is represented by a single point, which is helpful for
        cleaner visualizations in the map.
    Why:
        Clusters of hospitals or churches may result in overlapping or redundant data points, especially in dense
        areas. Reducing clusters by retaining only the most central point ensures the dataset remains representative
        without unnecessary duplication.

    How:
        The function creates a minimum bounding polygon for each cluster of points, which is then converted into a
        center point. The nearest hospital or church point to the center point is identified and retained, while
        the remaining points in the cluster are discarded. Hospital and church points that are not part of a cluster
        are merged with the retained cluster points. The 'RECTANGLE_BY_AREA' option is used to create the minimum
        bounding geometry for each cluster. The nearest point to the center of the bounding polygon is identified
        using a 'Near' analysis, and a dictionary is used to store the minimum distance values for each cluster.
        If two points have the same distance to the center, one is selected randomly. Finally, the non-clustered
        points are merged with the selected points from clusters, resulting in a merged feature class with reduced
        hospital and church points.
    """

    # List of hospital and church layers to merge at the end
    # Hospital and church points that are NOT in a cluster are already put in the list
    merge_hospitals_and_churches_list = [
        Building_N100.hospital_church_clusters___hospital_points_not_in_cluster___n100_building.value,
        Building_N100.hospital_church_clusters___church_points_not_in_cluster___n100_building.value,
    ]
```

generalization.n100.building.hospital_church_clusters.reducing_clusters() [source]

What:

Reduces hospital and church clusters by keeping only one point for each detected cluster. This ensures that spatial redundancy is minimized, and each cluster is represented by a single point, which is helpful for cleaner visualizations in the map.

Why:

Clusters of hospitals or churches may result in overlapping or redundant data points, especially in dense areas. Reducing clusters by retaining only the most central point ensures the dataset remains representative without unnecessary duplication.

How:

The function creates a minimum bounding polygon for each cluster of points, which is then converted into a center point. The nearest hospital or church point to the center point is identified and retained, while the remaining points in the cluster are discarded. Hospital and church points that are not part of a cluster are merged with the retained cluster points. The 'RECTANGLE_BY_AREA' option is used to create the minimum bounding geometry for each cluster. The nearest point to the center of the bounding polygon is identified using a 'Near' analysis, and a dictionary is used to store the minimum distance values for each cluster. If two points have the same distance to the center, one is selected randomly. Finally, the non-clustered points are merged with the selected points from clusters, resulting in a merged feature class with reduced hospital and church points.

GitHub Pages

SPHINX

GitHub Actions

Kartverket

Scale and maintainability

- Developing re-usable blocks
  - Saves time, most importantly in maintenance for refactoring etc.
  - But not everything (something is object/scale specific).
  - Based on centralized constants.

```
line_to_buffer_symbology = LineToBufferSymbology(
    input_road_lines=self.input_road_lines,
    sql_selection_query=self.sql_selection_query,
    output_road_buffer=self.output_road_buffer,
    root_file=self.root_file,
    buffer_factor=factor,
    fixed_buffer_addition=fixed_addition,
    keep_work_files=self.keep_work_files,
    write_work_files_to_memory=self.write_work_files_to_memory,
)
line_to_buffer_symbology.run()
```

```
building_polygons = PolygonProcessor(
    input_building_points=self.current_building_points,
    output_polygon_feature_class=self.output_building_points_to_polygon,
    building_symbol_dimensions=self.building_symbol_dimensions,
    symbol_field_name="symbol_val",
    index_field_name="OBJECTID",
)
building_polygons.run()
```

```
buffer_displacement_config = {
    "class": BufferDisplacement,
    "method": "run",
    "params": {
        "input_road_lines": ("roads", "input"),
        "input_building_points": ("building_points", "input"),
        "input_misc_objects": misc_objects,
        "output_building_points": ("building_points", "buffer_displacement"),
        "sql_selection_query": N100_SQLResources.road_symbology_size_sql_selection.value,
        "root_file": Building_N100.point_displacement_with_buffer___root_file___n100_building.value,
        "building_symbol_dimensions": N100_Symbology.building_symbol_dimensions.value,
        "buffer_displacement_meter": N100_Values.buffer_clearance_distance_m.value,
        "write_work_files_to_memory": False,
        "keep_work_files": False,
    },
}

buffer_displacement_partition_iteration = PartitionIterator(
    alias_path_data=inputs,
    alias_path_outputs=outputs,
    custom_functions=[buffer_displacement_config],
    root_file_partition_iterator=Building_N100.point_displacement_with_buffer___root_file___n100_building.value,
    dictionary_documentation_path=Building_N100.point_displacement_with_buffer___documentation___building_n100.value,
    feature_count="1400000",
)

buffer_displacement_partition_iteration.run()
```

```
point_displacement = BufferDisplacement(
    input_road_lines=Building_N100.data_preparation___unsplit_roads___n100_building.value,
    input_building_points=Building_N100.point_displacement_with_buffer___building_points_selection___n100_building.value,
    input_misc_objects=misc_objects,
    output_building_points=Building_N100.line_to_buffer_symbology___buffer_displaced_building_points___n100_building.value,
    sql_selection_query=N100_SQLResources.road_symbology_size_sql_selection.value,
    root_file=Building_N100.line_to_buffer_symbology___root_buffer_displaced___n100_building.value,
    building_symbol_dimensions=N100_Symbology.building_symbol_dimensions.value,
    buffer_displacement_meter=N100_Values.buffer_clearance_distance_m.value,
    write_work_files_to_memory=True,
    keep_work_files=False,
)
point_displacement.run()
```

Karty

Spørsmål?

Kontaktinfo

→ Ida Hope Barth

→ Ida.Hope.barth@kartverket.no

Kartverket


Kartverket