URBAN VISUALISATION & MANAGEMENT GMBH

UVM SYSTEMS

CityGRID® 2024

MANUAL

Builder

Content

Cover picture: Prague, Leonhard Niederwimmer, Pixabay

Note: This manual was produced by automatic translation. Strange expressions and sentence structure may therefore occur. In case of doubt, the German manual is to be used as a reference.

I. Basics CityGRID® Builder

1 *Definition CityGRID® Builder*

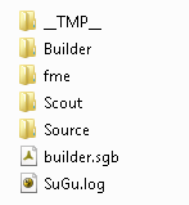
The City**GRID**® Builder is used for the compilation and optimization of processed 3D data for high-performance real-time visualization using City**GRID**® Scout. The Builder consists of different components that are either stand-alone applications or plug-ins in other software packages:

- a plug-in in Autodesk® 3D Studio Max, the "3D Studio Builder plug in"
- a plug-in in Safe Software FME, the writer "CityGRID Builder"
- the project tool "City**GRID**® Builder control center" for an overview and reorganization of data of a Scout Project (E.g. move to variants, deletion of data, etc.). Scouts can also be published (license unlocked) there.
- a command line tool running in the back ground, the "builder.exe"

The modules are partly based on each other and only partially interchangeable. Depending on the complexity and origin of the source data, different modules of the builder will be used to generate a 3D project. All kind of "static" data like point clouds, terrain models and City**GRID**® building models, can be efficiently prepared for the Scout via the FME Builder, whereas general 3D models, camera animations and animations of all kind have to be fed into the Scout project via the 3D Studio Builder.

Each data packet is created as a so-called Scene. The Scenes are unique within a Builder project and should always contain only data of one data class (see II 2.2). The Scenes are then resolved in the optimization of the Scout (see IV 5.1) and restructured into complex data trees. Scenes can be generated using both FME Builder and 3D Studio Builder and are treated in the same way as the Scout is optimized.

1.1 Construction of CityGRID® Builder projects



A City**GRID**® Builder project is composed of a collection of fixed predefined directories. Maintaining the directory structure is essential for the integrity of a Builder project. Therefore, manual intervention by the user should be avoided. When a new Builder project is created, the target directory has to be specified and the Builder acquires the directory structure independently.

Each Builder project has a file builder.sgb. This is the central management file for storage of global settings. Using this file, you can open the Builder project in Builder Control Center.

All log information from the Builder projects are included in the SuGu.log file. For support inquiries, deliver this file to UVM system.

Depending on the used plug in the folder structure can vary. The directories "Builder", "Scout" and "Source" are always available.

__TMP__: is a temporary directory for the creation of the Scout project.

Builder: This directory contains any scene using City**GRID**® 3D Studio Builder plug in. If the Builder project is loaded into 3D Studio, is this folder is evaluated. Each existing scene in 3D Studio is displayed in the Project Explorer.

ClusterDir: working directory used during combination of multiple Builder projects into clusters.

fme: This is a temporary working directory, used by the FME Builder plug in. Generally, this directory should be empty after successful processing in FME. Otherwise, a warning in the FME.log is found.

LinkDir: administration directory for storing the source folder of Builder projects.

ProcessDir: working directory that is filled with data from the Source folder during the optimization of a Scout project.

Scout: the final Scout project is stored in this directory after all processing and optimization work. Once the directory is filled with data, you can copy this directory to any other place, to allow the distribution of the Scout.

Source: All the processed data from the FME and 3D Studio Builder plug in are stored in this directory. As soon as all data for the Scout project have been prepared, the optimization of the Scouts from the Source data can be started in Builder control center.
Source data could be created gradually over any period of time. Mind that data will be overwritten if the same name is provided at iterative production cycles. Data in the Source folder will always be kept in full coordinates.

II. CityGRID® FME Builder

1 *Installation*

The FME Builder is technically seen a writer in FME, the data that is sent is divided into a specified folder structure and prepared for optimization of the Scout project. A supported version of FME has to be installed prior to the use of City**GRID**® FME Builder module.

<u>Note</u>: *The list of supported versions of FME can be found in the manual City**GRID**® basics. During the City**GRID**® setup all supported and installed FME versions found on the computer are selectable.*

<u>Note</u>: The City**GRID**® FME Builder is available from FME Base Edition but recommended is at least the use of FME Professional Edition.

At each change of FME version (e. g. upgrading to a new version) the City**GRID**® FME module component has to be un- and the reinstalled afterwards.

<u>Note</u> **:** Check change list of the delivered CityGRID® setup whether your installed FME version is supported by City**GRID**® and contact the City**GRID**® support in case of doubt.

2 *Description FME Builder*

When properly installed, the City**GRID**® Builder writer is listed at selectable FME Writers. If you add it to a workspace, a feature type is inserted, that can handle data of different kinds. To truly improve the performance of data processing and representation in City**GRID**® Scout the data is enriched attributive in an upstream Workspace (see II 2)

2.1 City GRID® Builder Parameter:

1. *Destination Folder* of writer is a directory in which a predefined sub-directory structure is created by the City**GRID**® Builder. These sub directories must not be tampered. You can also select the directory of an existing CityGRID® Builder project. This directory can be utilized both by FME Builder and the 3D Studio Builder together to compose a complex Scout project.

   <u>Note</u>: *Fanout dataset is not supported by City**GRID**® FME Builder.*

2. *Generate Scout* instructs the CityGRID® FME Builder to start the Builder Control Center immediately after finishing the FME workspace and to load the newly prepared Builder project so that the Scout can start optimizing directly.
3. *Instance Library* sets the path to a CityGRID® library, where prepared instance objects can be found for automatic placement in a CityGRID® Scout. Such libraries can be purchased from UVM Systems to display tree models or people in bulk in the Scout.

2.2 City GRID® Builder Attributes

In addition to the parameters determined, following standard attributes of City**GRID**® FME builder influence handling of sent data. Attributes that were generated in the course of an upstream workspaces are generally not evaluated and are ignored by the FME Builder.

<u>Note</u> **:** *The attributes of the FME builders are all optional and may be left unset. If one or more attributes are not set a preset default mechanism categorizes the data sent according to certain rules and classifies the data into the appropriate subdirectories.*

1. <u>cgbuilder_SceneID</u>:

   Each record has a unique ID assigned to the scenes that is used to uniquely identify the data in the builder project. This can have up to 8 digits, underscore or any special characters are not allowed.

Usage of scenes IDs recommended for visualization projects with many data sources in order to increase the clarity of the individual scene components and to simplify project management. If no ID is specified, "00" is used instead.

Note: The project code should be unique for the entire visualization project! If the sign " _ " is included in the project code, it will automatically be replaced by any valid character and issued a warning in FME .

2. cgbuilder_DatasetName:

   City**GRID**® Builder tries to give „meaningful“ names for the data of the Scout scene. If the attribute *cgbuilder_DatasetName* is set, this is used for the filename. Especially in more complex scenes, it is recommended to give meaningful names to individual data in order to identify them later. If the attribute is not set, the name for the data is set automatically. If the attribute *fme_dataset* is exposed at the Reader of the workspace, the file name of the read file is included.

3. cgbuilder_SceneName:

   To encapsulate data, the attribute *cgbuilder_SceneName* can be set. If a scene name assigned, then the processed data will be generated in the source directory of the project builder in a folder of the same name. Each folder which has been created may be used in Builder Control Center for defining variants (see IV 5.1.1). All data without set scene name will be associated with the main data set (eg. ambient model). These data are always visible in Scout and they can´t be assigned to a variant.

   Advice: Planning variants are used in order to change in real time without loss of performance between various planning states. For this purpose, the data of all variants must be preprocessed and then each stored as an independent planning variant. At runtime of Scout, all the data of the loaded variant will be discharged when switching to another variant and replaced by the data of the variant to be loaded. Data that should not be changed at runtime of Scout are called As built data and assigned to any planning variant.

4. cgbuilder_DataClass:

   Depending on the data class Builder applies different strategies to generate different "level of detail" ("LoD") for the Scout project and classify the data into a tree. This LoD system is independent of the LoDs of buildings and has significantly more levels. Depending on the distance of an object in the scene of Scout loads different LoDs to work performance optimized. Depending on the data class, thereby simplifying operations are made to the geometry or the imagery. It can also be objects hidden entirely. For a good performance of Scout for large data volumes therefore the proper recognition and indication of data class is essential. The attribute can have the following values (using either the name (case sensitive) or the numerical value is understood):

   - IMAGE

     is set for all the image data, which are sent via FME to City**GRID**® Builder. In particular georeferenced image data for texturing the terrain model (eg. orthophotos) are candidates for the assignment of this attribute.

   - STATICBUILDING

     All semantically correct City**GRID**® and CityGML buildings are assigned to this class. Compared to other 3D data, objects of this class of data are characterized by the fact that they occur separately in roof and façade (or any other elements) but may be combined to a building using an attributive description. In the derivation of LoDs the building is geometrically simplified more and more and can also be completely removed, in extreme cases (eg. As small objects at a great distance).

     Note: The characteristics of the data in the class Static Building must always correspond to a house character. The optimization algorithm used is optimized in this mold.

   - STATICTERRAIN

     Includes all 2.5D data which represent terrain or surface models. If textures are used, a regular mosaic of the same size as possible images (eg. orthophotos) should be used.

- STATICDESIGN

  All 3D data, which neither the Static Building nor the Static terrain data class are assigned, can be found here. Unlike the other two classes of data, Static Design type Data can not use the special optimization algorithm for the treatment of LoD levels, thus in very large scenes, the performance of the scene may be reduced. This is in turn ensures that all data, even in low LoD levels, are always present and never be removed.

  Note: All 3D data representing neither terrain models nor building shall be assigned to this category (roads, railways, monuments etc.).

- STATICPOINT

  All point clouds must be assigned to that data class by setting the attribute.

- STATICINSTANCE

  Recurring data of a 3D scene can be displayed optimized by means of instance objects (library objects). By combining an attributed point list from which the insertion points are obtained and the link to a City**GRID**® 3D library, the City**GRID**® FME Builder places the 3D objects in the Scout.

If the data type is not set, the following rules are applied:

- If the data comes from a City**GRID**®Reader, Builder can automatically detect the data class. There the reader attributes *citygrid_ElementClass* and *citygrid_ObjectClass* and *citygrid_BuildingElement_id* etc. may be used in particular.
- For semantically correct CityGML data the Builder may automatically detect the data class too. Hence CityGML data can be sent directly to Builder, in particular the LoD of the building model is used for optimizing Scout.

  *Advice: For very large amounts of data, creation of a City**GRID**® XML file with City**GRID**® FME Writer is advised as City**GRID**® data can even be better optimized.*

- For point data and image data, the data class is automatically detected from the geometry type.
- Instance data can not be detected automatically.
- Other data will be interpreted as "Static Design Data".

5. cgbuilder_OmitUntextured:

   This controls the behavior of terrain model surfaces without image overlap. Due to lack of coverage any texture can be applied, which can lead to unpleasant effects in the visualization. If features of digital terrain models have set attribute cgbuilder_OmitUntextured, the following system comes into force:

   - Value = 1:

     All untextured triangles are segregated and stored in a separate file in the "FME" directory of the project builder. For the creation of Scout only the textured parts of the terrain model are used, the untextured surfaces remain in fme folder for a possible detailed examination. In the log file of FME there is a warning that a file has been generated with all the untextured areas of the terrain model.

   - Value = 0

     or attribute does not exist: all triangles of the terrain model are sent to Scout, whether textures are assigned or not.

6. cginstance_Length:

   Defines the length of the instance object to be placed

7. cginstance_Width:

   Sets the width of the instance object

8. cginstance_Height:

   Determines the height of the instance object

9. cginstance_Diameter:

   For axisymmetric objects (such as trees), controls the diameter of the instance object

10. cginstance_Library:

    Defines the library name for the instance object

11. cginstance_Category:

    Sets the category name of an instance object

12. cginstance_Name:

    Specifies the name of the instance object to be placed.

*Example: In the City**GRID**® tree library named "Vegetation" lies in the category "Bionatics" the tree model "Maple". If the tree is to be 20m high and has a diameter of 8m, set the cginstance_Attribute as follows:*

cginstance_Height: 20

cginstance_Diameter: 8

cginstance_Library: Vegetation

cginstance_Category: Bionatics

cginstance_Name: Maple

Note: If no length, width, height, or diameter attributes are set, the Builder places the library objects with their original size of the library object in the Scout. Otherwise, the object is scaled accordingly. If only one value is defined, scaling is proportional, otherwise in each dimension specified.

When using the diameter parameter, only the height value must be set, since the diameter scales both the length and the width.

2.3 Tips for use of FME City GRID® Builder

In the following section, there are some typical questions and suggestions for the optimal use of City**GRID**® FME Builder.

2.3.1 Are there existing workspaces for data preparation?

For the most common data situations workspaces are provided in installation directory of City**GRID**® in subdirectories "FME Plugins\Builder" (for example, C:\Program Files\CityGRID\FME Plugins\Builder).

Note: The workspaces are overwritten with every setup. It is therefore advisable to create copies to a different location and to work with the copies. The workspaces in the installation directory shall be used as a template of the most recent state of the workspaces.

2.3.2 When should the Data Class be set?

While the data class for point and image data can be automatically detected from the geometry type, in surface data a distinction is only possible based on attributes. Therefore cgbuilder_DataClass should definitely be set for:

- Terrain models that do not come from a CityGRID or CityGML reader (for example, because they were generated by a Surface Modeller). Such generated models must be assigned to the data class STATICTERRAIN.
- Building data that does not come from a CityGRID or CityGML reader, has to be assigned to the data class STATIC DESIGN. In addition, the attribute citygrid_ElementClass should be set to "Roof", "Facade" or "floor" when the geometry for these elements is available separately. Thus, the standard color in Scout for roofs, facades and ground is set differently (see below) in untextured buildings.

2.3.3 What Types of Geometry are processed quickly?

Point cloud data should be defined as IFMEPointCloud.

Terrain models should be defined as IFMEMesh, based on triangles as geometry objects. City**GRID**® FME Builder can also handle any polygons in a mesh, but in that case, each polygon must be triangulated by City**GRID**® Builder again, which can be very time consuming, depending on the number of triangles.

Likewise, aggregates of all points or all terrain triangles (eg IFMECompositeSurface) can be used, but the processing time is higher in such cases.

2.3.4 Processing of CityGRID® Building Data

While City**GRID**® Writer has an extra feature for the unit (with the unit attributes) and the element (with the element-, element complex-, and object attributes), the builder has only a single Writer port. At this, primarily the surface data generated by City**GRID**® Readers will be sent.

*Note: Builder does not have the functionality to generate surfaces. For the triangulation of surfaces of a line structure which is present in City**GRID**® scheme, City**GRID**® Administrator, Modeler or FME Writer is necessary.*

For the handover of element attributes to Builder (eg the element class for automated colorization of the elements depending on the class), it is possible to transfer the attributes with help of Transformer FeatureMerger to the respective surface features. Similarly, the element features can be sent along with the surface features directly to the builder. In this case, the attributes *citygrid_Building*_id* as well as *cgbuilder_DataClass*, *cgbuilder_SceneName* and *cgbuilder_SceneName* have to be equal at all features.

*Advice: Sending the element features to FME Builder is recommended only if, in the course of an FME process just City**GRID**® data will be sent to the Builder. If different data sets are processed together with only one builder call, we recommend using a FeatureMerger for assigning all attributes to the surface features.*

Note: line data need not be sent to Builder, since they are not visualized in Scout anyway.

2.3.5 CityGML Building Data

Semantically correct CityGML data is automatically interpreted in best possible way by City**GRID**® Builder. It should be noted:

- If the CityGML data include a Terrain Intersection Curve (TIC), the processing in FME can lead to (unwanted) generating a surface, which has visual impairments in Scout result. To prevent this, it is advisable, not to send the TIC to FME Builder.
- The FME CityGML reader reads always only the first texture theme and ignores any further. Texture that is not addressed with the first texture them, is currently ignored in FME, and therefore also ignored at Scout projects.

2.3.6 Any Building Data with separate Elements

If building data do not originate from City**GRID**® or CityGML, there are no general rules about which areas can be combined into a building or into an element (roof, facade). The user has to do this, by grouping and setting attributes in the workspace. For this purpose, the user should generate the following attributes:

- *citygrid_DataClass* with value STATICBUILDING
- *citygrid_FeatureType*: with the value "Surface" to tell Builder that these data are faces of a building part.
- *citygrid*_BuildingUnit_id, *citygrid*_BuildingObject_id, *citygrid*_BuildingElementComplex_id and *citygrid*_Element_id to be generated with unique values depending on the complex in order to simulate the City**GRID**® Building hierarchy.

Example: Four Building IDs to be assigned to a building with a main object and a garage. Therefore, the following attribution is made:

To all features of the building citygrid_BuildingUnit_id 1 is assigned, since they are all part of the building.

*All features of the main building form an object, according to City**GRID**® data structure, and therefore receive the citygrid_BuildingObject_id 1. All features of Garage get the value 2.*

As the building is not further subdivided (eg in elevator object, ramp etc), the citygrid_BuildingObject_id may be equal to the citygridBuildingElementComplex_id

The individual surfaces of the main building should be structured into Roof, Façade and Floor (if there is ever a floor). The citygrid_BuildingElement_id attribute shall be assigned to the features of each Element class, with the values 1, 2 or 3.

This results in all features of the facades of the main building having a sequence of attributes 1/1/1/2 respectively the bottom of the garage object having 1/2/2/3.

Note: Depending on the data, generating the mentioned citygridBuilding.ids is not always possible, or it is only possible to a certain hierarchy level. It is recommended to always structure the data as good as possible in order to support the preparation of visualization LoDs best. The Scout project is also generated if the external data is not structured hierarchically. However, performance problems are more likely the less structured the data, which are sent to Scout.

- *citygrid_ElementClass* at „Roof“, „Facade“ or „Floor“, if standard colors to be assigned depending on the element class.

2.3.7 Is it possible to process any Number of DTM tiles?

Like any computer process, working with City**GRID**® Builder is bound to memory limitations. The maximum amount of data which can be processed depends on the computer configuration. Once the memory limits are exceeded, however, it comes to cancel the operation. This is generally recognized by so-called "Bad allocation" errors. To prevent this buffer overflow, you only have the option of dividing the input data on multiple processes. This for the time being results in several scenes, each having an optimized data structure. The final Scout can then be composed of these scenes.

Advice: For large datasets, it is advisable to proceed iteratively and first editing several creation processes with each having small amount of data. By successively increasing the amount of data, it is possible to evaluate the maximum ammount of data precessable using the available computer configuration.

2.3.8 Will Image Boundaries of a Terrain Texture be generated automatically?

Terrain texture can be used by both City**GRID**® Reader, as well as Tiff or Jpg or similar reader (if a world file exists). The Builder automatically calculates image borders into the triangulation. To obtain the full sharpness of terrain texture, images may have a maximum of 1024x1024 pixels. This can be achieved with preprocessing using RasterTiler in FME.

2.3.9 Which geometry data can be processed with instance objects?

Instance objects need an insertion point to be placed in the Scout. This insertion point is extracted from point features that are sent to the City**GRID**® FME Builder. Other data classes as points can not be used for instance objects. If the points have 3D coordinates, the writer takes these values and places the objects accordingly. If there are no 3D values, 2D point features are to be sent to the writer, the instance objects are then placed at height 0. Using the Builder Control Center, it is possible to change the height of instance objects by elevating the points on surfaces of other scout scenes (see IV 5.3).

2.3.10 Can self-created library objects be placed via FME?

This is not possible because individually created objects are saved as .max files in the library (see III 4.3). The proprietary .max format can not be processed using FME.

2.4 Creating Builder projects using FME workspaces

UVM Systems provides a set of predefined workspaces for the creation of builder projects, which can be used to process the most common data types. To use the workspaces, the corresponding published parameters must be filled with values; an adaptation of the program logic is generally not necessary. Any customization work is

offered by UVM Systems as part of CityGRID® support at the customer's request. If necessary, please contact CityGRID®Support.

Note: The workspaces below are part of the CityGRID® setup and are copied in the installation directory under FME Plugins \ Builder. These workspaces should be used as templates. Editing in the installation directory is not useful and should be avoided.

2.4.1 Builder_CGUnits

The task of this workspace is to convert CityGRID® units into a builder project. Only the areas of the units are evaluated, the line structure cannot be processed in Scout and is therefore not evaluated by the workspace. The workspace does not make any changes to the faces, so the units must be brought into the desired shape by means of appropriate preprocessing steps.

Note: The units must have triangulated faces in order to be processed. It is therefore irrelevant whether it concerns units with a linear structure or face based data with the face generation type "frozen face net".

The following parameters must be set for correct operation. The values in [] represent the parameters for batch operation of the workspace:

- *tile: [p_TILE]*

  Optional parameter to be able to set recurring names, such as the tile ID of a map tile, as a variable. This value can be used as part of other parameters of the workspace. To be able to access the value, the reference "$(p_Tile)"must be used.

- *Modelname (separated by ';')*: [MODEL_NAMES]

  Optional parameter to be able to load models from a CityGRID® database. In contrast to XML files, which can only ever contain a single model, databases can have any number of models. By specifying the model name, the CityGRID® FME Reader is instructed to load the exact model from the data source. When using XML files as a data source, this parameter should remain empty.

- *Input CityGRID building: [S_CITYGRID*]

  Mandatory parameters with the path to CityGRID® XML file (s) with units- Alternatively, the path to a CityGRID®-ini file with the login parameters to a CityGRID® database can be used. If several XML files are to be read, the wildcard "*" can be used instead of the file name.

- *Use textures: [READ_IMAGES]*

  Mandatory parameters for the selection of the surface quality. The following values can be set:

  - yes:

    The textures of units are forwarded to the builder project and used in the creation of the scout. CityGRID® standard colors are given to any untextured surfaces

  - no:

    Any existing textures are not read out and the surfaces of the units are colored according to the parameters of the "Surface colors" parameter.

- *Directory: builder project: [D_CITYGRIDBUILDER]*

  Mandatory parameters to define the location of the builder project. The CityGRID® FME Builder creates the folder structure of the Builder project under this path (see I 2) and saves the data prepared by the workspace. This directory also contains the builder.sgp file, which can be used to load the project in the Builder Control Center to create the Scout. (see VI 3)

- *Face colors: [p_COLOR]*

  Mandatory parameters for the selection of the surface coloring if no textures of units are to be read out. The following values can be set:

  - CG standard:

Any existing textures are not read out and the surfaces of the units are colored with the CityGRID® standard colors.

- Individual:

  Any existing textures are not read out and the areas of the units are colored with individually set color values. A distinction is made between roof areas and all other areas. The color for the roof and the other areas, summarized under "facades", can be set in the private parameters of the workspace.

- Unit Color:

  Any existing textures are not read out and the surfaces of the units are given a uniform color. The uniform color can be set in the private parameters of the workspace.

- *scene: [p_SCENE_NAME]*

  Mandatory parameters to be able to address the prepared data package within a builder project. Data of a scene represent related data of a data class (e.g. all units of a sheet cutting cell). This data package can be evaluated in the Builder Control Center, during the create process and included / excluded in the scout creation.

  Note: Scenes are used to organize the raw Builder data. The information about the scene is lost during the creation of the scout

- *File name: [p_DATASET_NAME]*

  Mandatory parameters for designating the processed files in the Builder project. The file name must be set individually for each scene to prevent overwriting of data that has already been processed.

  Advice: The optional parameter Tile can be used for the parameters scene and file name. By setting $ (p_Tile) for scene and file name, the value of tile is transferred. In this way, you only need to change the value of the tile to ensure that the scene and file name are set with the same, unique values.

- *Log file: [LOG_FILE]*

  Mandatory parameters to specify the location of the FME log file. Please note that the path to the log file must already exist, FME does not create any missing subdirectories of a manually set log path.

2.4.2 Builder_DGM

The task of this workspace is to transfer terrain models into a Builder project. The workspace accepts both terrain models in CityGRID® data format and terrain data in DWG, DGN, Shapefile, ASCIIGrid or FFS formats as input. In addition, the workspace offers the option of texturing the terrain data using georeferenced raster data (Worldfile).

Note CityGRID® terrain models, as well as all other data that are already brought into the workspace as a TIN surface, can be processed directly. The workspace triangulates all other data, which is then passed on to the Builder.

The following parameters must be set for correct operation. The values in [] represent the parameters for batch operation of the workspace:

- *tile: [p_TILE]*

  Optional parameter to be able to set recurring names, such as the tile ID of a maptile, as a variable. This value can be used as part of other parameters of the workspace. To be able to access the value, the reference "$ (p_Tile)" must be used.

- *Input DTM: [S_TIN]*

  Mandatory parameter that contains the path to the terrain model base data.

- *Format DTM: [S_FORMAT_TIN]*

  Mandatory parameter that determines the file format of the previously imported terrain model output data. The data can only be read in if the combination of both parameters is set correctly.

- *Input terrain texture: [S_IMAGE]*

Mandatory parameters to define the path to texture images for the terrain texture. If several images are to be used in a directory, the wildcard "*" can be used as a wild card.

Note If no terrain texture is desired, set an invalid path, e.g. "not used". The workspace can only be started if this parameter has been loaded with a string.

- *Format terrain texture: [S_FORMAT_IMAGE]*

  Mandatory parameter that determines the file format of the previously read terrain texture. The data can only be read in if the combination of both parameters is set correctly.

- *Insert point Worldfile: [p_INSERT_POINT]*

  Mandatory parameters for the location of the insertion point are defined in the georeferencing file of the image (Worldfile). By default, the insertion point is always at the center of the top left cell. In the case of a non-standard world file, the insertion point can also be set to the corner point. The workspace then automatically corrects the position by half a ground pixel.

  Note With the coordinates of the world file, it must be ensured that both the resolution of the pixel size on the ground and the coordinates of the insertion point are given to exactly three decimal places. If necessary, the image must be resampled in the workspace or the position adjusted.

  If the terrain texture is derived from database without map tiles, the section should always be cut exact to the meter. The free drawing of an area of interest generally does not lead to the required coordinate sharpness at the point of insertion. When processing such data, a large number of untextured triangular surfaces appear at the edges because the texture coordinates cannot be determined exactly.

- *Directory Builder project: [D_CITYGRIDBUILDER]*

  Mandatory parameters to define the location of the Builder project. The CityGRID® FME Builder creates the folder structure of the Builder project under this path (see I 2) and saves the data prepared by the workspace. This directory also contains the builder.sgp file, which can be used to load the project in the Builder Control Center to create the Scout. (IV.3)

- *scene: [p_SCENE_NAME]*

  Mandatory parameters to be able to address the prepared data package within a Builder project. Scene data represent related data of a data class (e.g. all terrain models of a map tile). This data package can be evaluated during the creation process in the Builder Control Center and in- or excluded in the Scout creation.

  Note If no terrain texture is desired, set an invalid path, e.g. "not used". The workspace can only be started if this parameter has been loaded with a string.

- *File name: [p_DATASET_NAME]*

  Mandatory parameters for designating the processed files in the Builder project. The file name must be set individually for each scene to prevent overwriting of data that has already been processed.

  Advice: The optional parameter Tile can be used for the parameters scene and file name. By setting $ (p_Tile) for scene and file name, the value of tile is transferred. In this way, you only need to change the value of the tile to ensure that the scene and file name are set with the same, unique values.

- *Log file: [LOG_FILE]*

  Mandatory parameters to specify the location of the FME log file. Please note that the path to the log file must already exist, FME does not create any missing subdirectories of a manually set log path.

2.4.3 Builder_Pointcloud

The task of this workspace is to transfer point cloud data into a builder project. The workspace only accepts LAS or LAZ data as input. In addition, the workspace offers the option of coloring the point clouds using georeferenced raster data (world file).

Note: Point clouds can only be displayed if the point distribution corresponds to a currently used ALS flight. Dot densities of up to 60 dots / m² can be displayed without any problems and the data volume is limited only by the hard disk capacity. Point clouds from short-range scans, however, are not suitable for processing because the point density is too high.

The following parameters must be set for correct operation. The values in [] represent the parameters for batch operation of the workspace:

- *tile: [p_TILE]*

  Optional parameter to be able to set recurring names, such as the tile ID of a maptile, as a variable. This value can be used as part of other parameters of the workspace. To be able to access the value, the reference "$ (p_Tile)" must be used.

  Point clouds LAS file (s): [S_LAS]

  Mandatory parameters for the location of the point cloud data. Data in the formats LAS and LAZ (packed LAS) can be processed. If several point cloud files are to be used in a directory, the wildcard "*" can be used as a wild card.

- *Input ortho photo: [S_ORTHO]*

  Mandatory parameters to determine the path to texture images for the coloring of the point cloud data. If several images are to be used in a directory, the wildcard "*" can be used as a wild card.

  Note: If no coloring is desired, then an invalid path, e.g. To set "not used". The workspace can only be started if this parameter has been loaded with a string.

- *Format of terrain texture: [S_FORMAT_ORTHO]*

  Mandatory parameter that determines the file format of the previously imported orthophotos. The data can only be read in if the combination of both parameters is set correctly.

- *Directory Builder project: [D_CITYGRIDBUILDER]*

  Mandatory parameters to define the location of the Builder project. The CityGRID® FME Builder creates the folder structure of the Builder project under this path (see I 2) and saves the data prepared by the workspace. This directory also contains the builder.sgp file, which can be used to load the project in the Builder Control Center to create the Scout. (see IV 3)

- *scene: [p_SCENE_NAME]*

  Mandatory parameters to be able to address the prepared data package within a builder project. Data of a scene represent related data of a data class (e.g. all point clouds of a maptile). This data package can be evaluated during the create process in the Builder Control Center and included in the scout creation.

  Note If no terrain texture is desired, set an invalid path, e.g. "not used". The workspace can only be started if this parameter has been loaded with a string.

- *File name: [p_DATASET_NAME]*

  Mandatory parameters for designating the processed files in the Builder project. The file name must be set individually for each scene to prevent overwriting of data that has already been processed.

  Advice: The optional parameter Tile can be used for the parameters scene and file name. By setting $ (p_Tile) for scene and file name, the value of tile is transferred. In this way, you only need to change the value of the tile to ensure that the scene and file name are set with the same, unique values.

- *Log file: [LOG_FILE]*

  Mandatory parameters to specify the location of the FME log file. Please note that the path to the log file must already exist, FME does not create any missing subdirectories of a manually set log path.

III. CityGRID®3D Studio Builder Plug In

1 *Installation and Start*

The 3D Studio Builder Plug In is, technically speaking, an extension of the functionality of Autodesk® 3D Studio Max. With help of 3D Studio Builder Plug In, Scout projects may be processed by arranging of 3D data, generation of camera paths and animations. Data from 3D Studio Builder Plug In is structured into a predetermined folder structure and prepared for the optimization of the Scout project. As a prerequisite for the use of 3D Studio Builder Plug In, a supported 3D Studio version has to be installed on the computer before installing the City**GRID**® module.

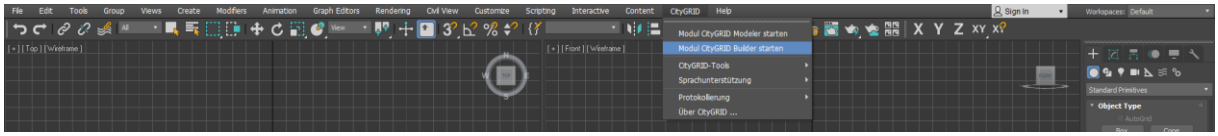
*<u>Note:</u> The list of supported 3D Studio versions can be found in the manual City**GRID**® Basics. During the City**GRID**® setup process, all Studio versions can be selected that are installed on the computer and are supported by City**GRID**®.*

*Note: City**GRID**® 3D Studio Builder Plug In is available in the basic version of 3D Studio as well as in the design variant.*

At each change of version of 3D Studio (such as when upgrading to a new annual version) the entire component of Autodesk City**GRID**® must be uninstalled and reinstalled after reinstalling 3D Studio.

*<u>Note:</u> Check the list of changes delivered together with the City**GRID**® setup package whether your 3D Studio version is also supported by City**GRID**® and contact the City**GRID**® support in case of doubt.*

After successful installation and licensing (see Manual City**GRID**® basics) the menu item **CityGRID** appears in the main menu bar of 3D Studio, with a mouse click on this menu item appears **Start CityGRID Builder Plugin**.



1.1 Setting Language

Depending on which language is currently active, in the menu **CityGRID** the item **Localisation**, or **Sprachunterstützung** is available. Select the preferred language of the user interface.

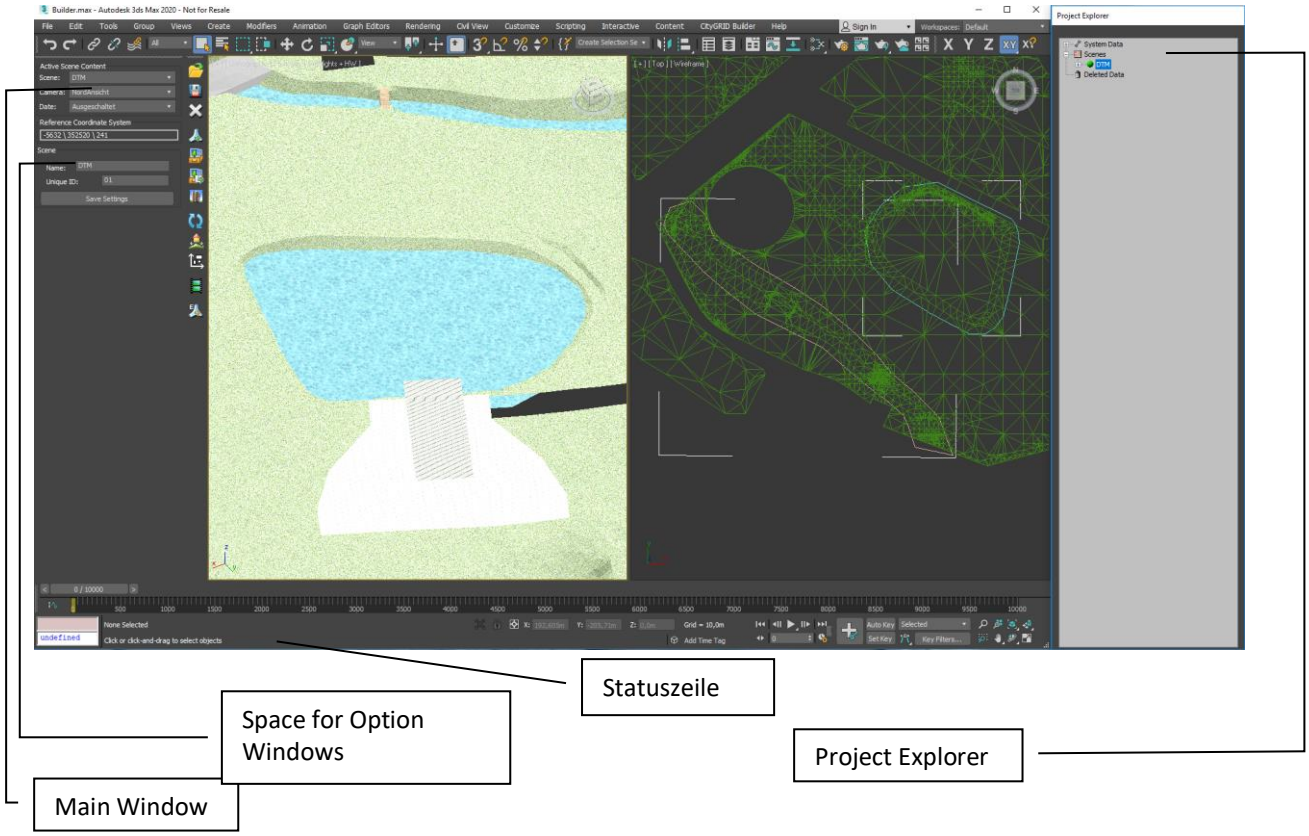
1.2 Starting the City GRID® 3D Studio Builder

As soon as the menu item **Start CityGRID Builder Plugin** is clicked, the user interface of 3D Studio is adjusted and Builder Plug Inwill be started. Immediately after the start i the Project Settings window (see III 3.1.1) appears. Select either an existing Builder project or create a new project.

<u>Note:</u> 3D Studio Builder Plug In can only be operated in combination with a valid Builder project. Either you have to create a new project or load an already existing one. Builder Plug In will terminate, if no selection is made in the Project Settings window.

2 *The graphic user interface*

After having started the planner via menu item **CityGRID – Start CityGRID Builder Plugin** the user surface will be adapted. Menu item **CityGRID** will be replaced by menu item **CityGRID Builder**. Additional windows will be docked (the main window of the Builder and Scene explorer) and several additional buttons (the main toolbar) will appear. In addition, the window *Create/load Project* will be opened.



The main window consists of the quick selection groups planning, camera, sun position, perspective control and the display of the reduction factors of the local coordinate system. (see II 2.2).

The Project Explorer contains several project data: geometrical data of stock and planning, as well as system data, like sun position or camera angles. (see II 2.3)

Functions are either started via the main toolbar or via the builder menu. Optionally appearing options windows are adapted to the Project Explorer below.

If a user-action is expected, an according notification is shown in the status bar.

For a new project two windows will be prepared: a ground plan window on the left and a perspective 3D view. The configuration of the window can be adapted by the user and is saved together with the project.

The Command Panel of Autodesk 3D Studio MAX is blanked out by default, since mainly functions of the City**GRID**® Plugins should be used for designing a scene. Command Panel could be optionally turned on using right click next to the **Main Toolbar** and toggle **Command Panel**.

2.1 Toolbar buttons

Create New City**GRID**® Builder Project

Save project

Close City**GRID**® Builder plug-in

Load City**GRID**® Data

Add Library Objects

Set Library Objects with External Data File

Import External Data

Refresh View

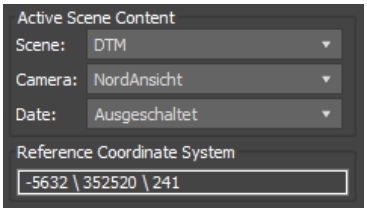
Place Objects on Terrain

Duplicate Objects

Animationen und Bilder rendern

Export Source Data for City**GRID**® Scout

2.2 Main Window

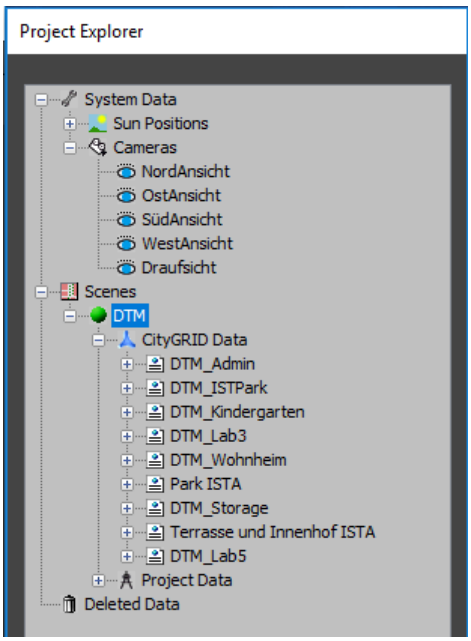


The main window contains fields for the quick selection:

- current scene (see III 3.2)
- active camera (see III 5.1)
- date for Sun position (see III 5.2)

Finally, the reduction constants of the transformation of the global coordinate system are shown in the local coordinate system shown in 3D Studio MAX. (see III 0)

2.3 Project Explorer



Project Explorer consists of several project data in a tree-display. The following main nodes can be found in the Project Explorer:

- System Data:

  such as project-oriented system data, like various sun positions- and camera angles.
- Scenes:

  various scenes, which consist of their own geometrical data. There is always a current (loaded) scene, marked with a green dot.
- Deleted Data:

  deleted data is removed to Deleted Data. As long as this folder is not deleted, data can be restored.

Note: To permanently delete the contents of the Recycle Bin, the menu item "CityGRID Builder" of the 3D Studio Max main menu bar provides the function "Clear Deleted Data". By executing this menu item, the contents of the Recycle Bin are emptied in the Project Explorer and the associated backup files in the Builder project are deleted.

- Selection :

  All currently selected elements in viewport are grouped here and can be moved together.

By selecting a node in the Project Explorer, the according Options window (if available) as well as the export settings for the City**GRID**® Scout (see III 8.1) are docked onto the *Project Explorer*.

2.4 Drag&Drop

Nodes in the Scene Explorer can be dragged and dropped (apart from main nodes System Data, Planned Data, As-Built Data and Deleted Data). Furthermore, the current active sun position, the active camera as well as the loaded Scene are not available for drag & drop. The rest can be operated via drag & drop anytime and is immediately shown in the Scene Explorer. For example, As-Built Data can be dragged into a Scene.

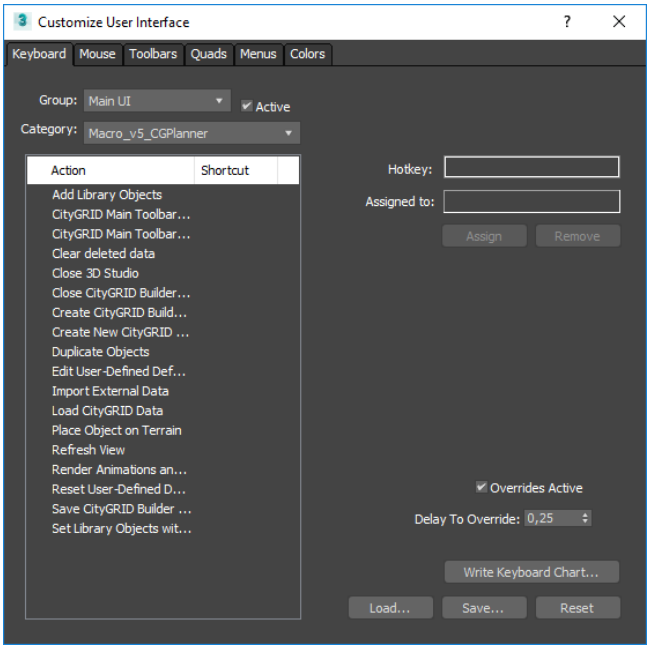
1. Click on the nodes in the Scene Explorer with the left mouse button, which represents the data to be dragged.
2. Keep the mouse button pressed and drag the mouse over the node in the Scene Explorer, which the data should be dragged to. When a node, which the data can be dragged to, is reached, the mouse pointer changes.
3. When the aimed node in the Scene Explorer is reached, release the mouse button.

Advice: In order to make multiple choices relocatable via drag & drop, the entry Selection in the Scene Explorer was created. It has effect on all touchable objects of the planner project and appears immediately after an object in Viewport, or a sheet (the lowest level in the scene explorer seen from a hierarchical point of view), is clicked in the Scene Explorer. Multiple choices are only possible via a selection in Viewport. The dragging of the Selection node in the Scene Explorer has simultaneous effect on all chosen objects.

2.5 Assigning Hotkeys in Autodesk 3D Studio Max

A number of actions of the City**GRID**® Builder can be started in addition to clicking at the respective buttons by user defined hotkeys. Assigning hotkeys is done by opening the dialog **Customize > Customize User Interface.** At the tab *Keyboard*, for the group *Main UI"*, the category *Macro_v5_CGPlanner* can be selected.

Each of the following actions of the City**GRID**® Builder can be assigned a hot key. By clicking on `Save` all assigned hotkeys and the current configuration of the 3D Studio user interface is stored. For example, the position of toolbars can be individually defined this way.



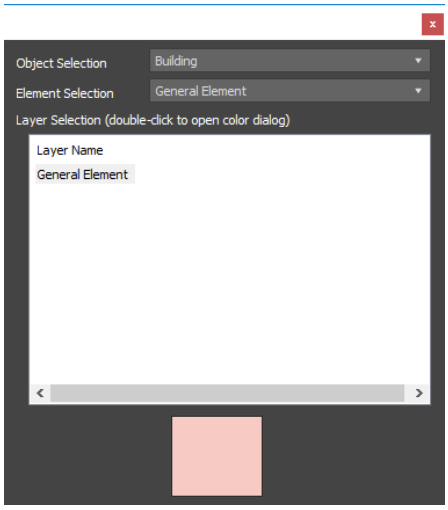
2.6 Changing Colors of CityGRID® data

Every default color City**GRID**® utilizes can be altered on demand.

*Note: Changes made in City**GRID®** Builder to the color scheme will also have effects in City**GRID®** Modeler and vice versa.*

The user defined default colors of line and face layers of Units can be changed by opening the menu City**GRID**® **Builder > Edit User Defined Default Colors**…. A dialog is opened for setting the layer colors individually for each Object selection and Sub-Element selection.

The color selection dialog can be opened by double-click at the respective layer name or by clicking at the color field.



3 *Working with the CityGRID® 3D Studio Builder*

One of the main functions of the City**GRID**® Builder is the assembly of visualizations with data from various sources as well as the processing of various Scenes. Geometrical data can originate from the following sources:

- City**GRID**® Data :

  3D-models of buildings and other 3D objects (walls, bridges, etc.), which are administered in a City**GRID**® data bank, can be taken into the project (see III 3.3). These data mostly make the actual stock.

- Project data :

  3D-models of designed buildings and other 3D objects, which either are imported from external files (e.g. files created by architects) or operations are carried out directly in the project via 3D-Studio (see III 4.2).

- Library data :

  Parameterised 3D-models, which are created from library models (with parameters, such as width, length, height, etc.; see III 4.3);

3.1 Projects

A City**GRID**® Builder Project consists of a row of Subdirectories (see I 0). One of them is named "Builder". All the scenes of the project are saved in it.

__Projects__: all the scenes are saved here.

__Media__: pictures and videos are produced here, which are rendered in the Planner.

__Textures__: textures are stored here, which are created in the Planner (e.g. the texture picture for the area section).

__Deleted__: Files from recycle bin can be found here.

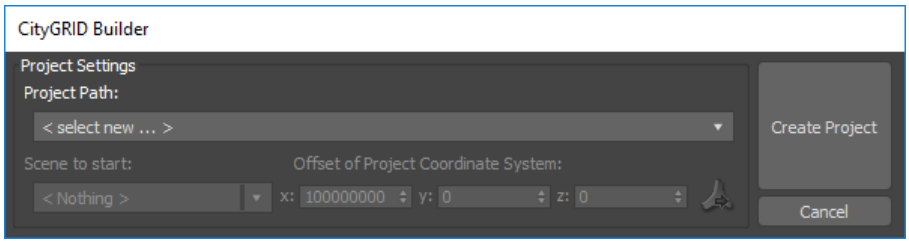
Note: The directories listed represent only the data of 3D Studio Builder and do not represent a complete Builder project. A complete Builder project is only represented in the case of the directory structure as described under 2.

In the case of support requests to UVM system you should always send the entire project builder when prompted!

3.1.1 Create New CityGRID® Builder Project

The window for creating a new City**GRID**® Planner project is automatically opened after the launch of the module.

The window can also be opened later via the button `Create New Project`.



1. *Project Path*:

   In the selection list, an existing Builder project can be selected, or a new project can be created by clicking on *select new.*

   Advice: Any existing projects are listed both with the project name and the location.

   If an existing project is selected, the other parameters of *Project Settings* will be set automatically.

   When creating a new project, a file browser opens and the location of the project builder has to be specified. With help of standard Windows functions, new directories can be created in the browser. Once a directory has been selected (see III 2) the Builder directory structure will be created. By this, the new project will be initialised.

   Advice: Builder projects may contain huge amounts of data. When defining the location, you should pay attention to adequate space. During processing, in particular during the preparation of the Scout, it is advisable to work on a local hard drive. This is because, in the course of this step, a large number of files are stored on the hard disk space. When working on a network, this leads to unnecessary traffic and sigificantly reduced performance.

2. Scene to start:

   If the Builder project consists of several scenes, one of those scenes can be selected and loaded into the 3D Studio with help of this selection list. All other scenes are be recorded as entries in the Project Explorer and can be changed during operation with the 3D Studio Builder. In a newly created Builder project, selecting the start project is not available because no data is available.

3. Project Coordinate Offset:

   The generation of scenes in 3D Studio Builder is carried out using a local project coordinate system. Since 3D Studio does not provide sufficient precision, City**GRID**® shifts the coordinates of City**GRID**® buildings loaded to such a local coordinate system having reduced coordinates. Once the data is processed and stored in the source folder, the project coordinates offset is reattached, so the data is available again in the initial coordinate system.

   The project coordinates offset is generally indicated for the whole Builder project and applies to any scene of the Builder project which is produced with 3D Studio. A subsequent change of the offset is not possible.

   Optionally, an offset value can be specified by the user, namely by specifying shift values for x, y and z are given. The values given are deducted from the initial coordinates to define the local project coordinate system Optionally, with help of the button `Get offset of project coordinate offset from CityGRID Data` a City**GRID**® model may be selected, which is used for automatic evaluation of shift values.

   *Note: In the Scout Scene to be created, external data (eg, architectural models) should properly fit to City**GRID**® buildings. For that reason, external data should be transformed into the project coordinate system before import into Builder.*

4. Open Project

By clicking the button Create Project thebuilder project is loaded. If this is a new Builder project, the following actions are performed:

- In the file system a new Builder project is created with the pre-defined folder structure.
- The veiwports are reset by default (a ground viewport and a 3D viewport).
- A scene called “Standard” is generated.
- Five standardized camera settings type “relative camera” (see III 5.1) are generated: North-, East-, South- and West Elevation and Plan.
- Furthermore, eight standardized Sun Positions (see III 5.2) are generated: the 4 seasons change each with a fixed time at noon and a time from morning till night. Here, the specified time of installation site is used by default.to the four-seasons-change each with a fixed point of time at noon and a period of time from the morning to the. Here, the specified location during installation is used by default.

Afterwards the user can add further data (see III 4)

3.1.2 Save Project

The project is automatically buffered several times. Especially before skipping to another Scene, the current Scene is saved.

In addition, the project should be saved regularly by clicking the button of the main toolbar `Save CityGRID Builder Project`. By doing so, various data as well as the view settings are saved.

<u>Advice</u>: To avoid time consuming saves, always run a Builder project on a local drive. Due to network restrictions (traffic and response times) usage of network drives is not recommended.

3.2 Scenes

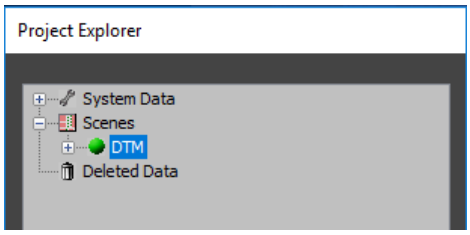
Each Builder project consists of at least one, but often several scenes. Since Builder projects often are large and detailed compositions from different 3D data, it is advised to divide the project into several partial sections, so to decompose into several smaller scenes. In this way, the entire Builder project can be better reworked, it is more clearly arranged and can be updated easily. Each scene contains a specific portion of the total project builder and the sum of all scenes constitutes the Builder project. The individual scenes can be created both on the 3D Studio Builder, as well as via the FME Builder.

<u>Note:</u> If the project is very small, if necessary, the entire project can be encapsulated in one scene.

In 3D Studio any number of scenes can be created, but only one scene can be loaded into the viewport. All other scenes are stored in the appropriate directory of the builder project and unloaded from 3D Studio. By selecting some other scene in the Project Explorer, it is possible to switch between alternative scenes. The active scene is unloaded from 3D Studio and the selected scene imported.

3.2.1 Create new Scene

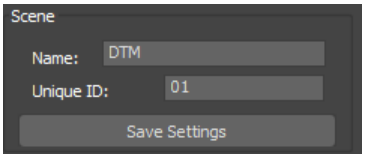
Like any other new component in 3D Studio Builder, also scenes are generated by selecting the main node in the project explorer created. By clicking at the scene node, a property window opens. There, the scene name and a project ID must be entered.



- *Name*

  Enter the new scene a unique *Name*
- *Unique ID*

Unique ID Unique ID is used to assign a unique identifier to the scene, in addition to the scene name. This parameter is used to manage the scene. The scene ID must be 8 characters long and contain both characters and digits. If an already assigned ID is set again, an error message appears.



Note: The scene ID also offers the possibility of project administration by the user. The introduction of a label scheme provides a better overview of different scenes with large Scout projects. For example, a labeling scheme of the first character could be like this: 1 = terrain model (StaticTerrain), 2 = building model (StaticBuilding), 3 = general 3D data (StaticDesign), 4 = pointcloud (Static Point) 5 = Instance objects (StaticInstance). The scene ID is valid for the entire Builder project and can therefore be set from 3D Studio and FME for each scene individually.

- *Create*

  By selecting `Create` the new scene is created. Then the scene is added and activated in the Project Explorer under the node scene. Subsequently, data can now be inserted in this scene.

3.2.2 Change to the active Scene

In a City**GRID**® Builder Project always one Scene is the active one. It is the one whose data is shown. Non-active Scenes are unloaded by the 3D Studio MAX in order to save disk space.

The current Scene is marked with a green dot in the Scene Explorer and can either be selected in the Scene Explorer or via the quick selection in the main window. Since other Scenes are unloaded, the contents of a Scene can only be unfolded for the active one.

*Note: When changing the active Scene the City**GRID**® Builder Project is automatically saved, since the last active Scene is unloaded.*

3.3 Requirements for 3D Objects for CityGRID® Scout

3D objects must satisfy the following conditions in order to be properly visualized in City**GRID**® Scout.

All City**GRID**® geometry objects satisfy the following conditions by default. However, if the user creates his own library objects (see III 4.3), or integrates external models (see III 4.2.1), he has to pay attention to the following conditions on his own.

3.3.1 Geometry

Merely geometry objects of 3D Studio types „`Mesh`“, „`Trimesh`“ or „`EditableMesh`“, may be used, or those geometry objects, which are convertible into these three mesh types. It should be noted that geometry might be varied in the course of the conversion, for example 3D Studio might alter to freeform surfaces (`Patches`) during conversion.

For efficient surface representation City**GRID**® Scout uses only the normal vectors of 3D surfaces. A double-side representation is not available by default! Optional display variants in 3D Studio will be ignored when exporting to City**GRID**® Scout (eg, double-sided surface representation, material type single-sided surface representation etc.)

3.3.2 Material

It supports merely standard material and multi-material (consisting of standard materials). Just "`Diffuse Color`" and "`Diffuse Map`" are evaluated from the standard material, all other "`Maps`" are ignored. For example, architectural materials can not be adopted. From "`Diffuse Map`" merely the map types "`Bitmap`" and "`bitmap 2D`" are supported. `Cropping` and `Placement` for bitmap may not be used in the material editor! An application of `Cropping/Placement` causes failure of the 3D object inCity**GRID**® Scout.

4 Work with geometrical data in CityGRID® Builder Project

4.1 Load CityGRID® Data

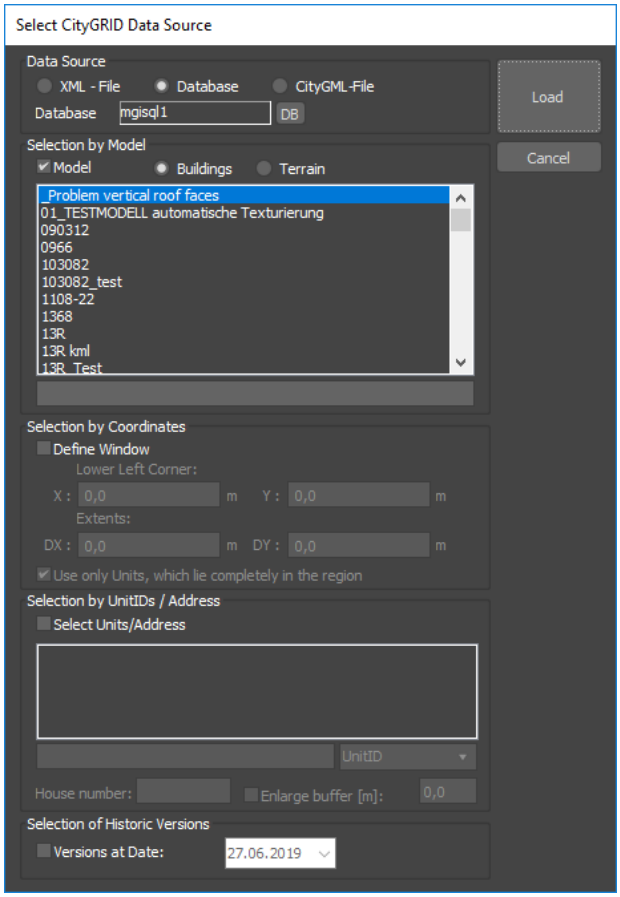
At the beginning there is the definition of the working area and the data source.

4.1.1 Open/change Database connection

1. The button `Load CityGRID Data` opens the pop-up window.
2. Select *Database* in section *Data Source*.
3. Open *Database Settings* window with the DB button.

Type in name of the installed datasystem, server, scheme, database, user and password. (Ask the database administrator for this data).

Hinweis: Note: Valid settings are saved and can be loaded later easily via the selection "Saved services". By storing the server and database name / service more than one database can be accessed quickly when using a MSSQL Server. When you start the modeler is always trying to re-establish the last active database connection.



4.1.2 Open/change XML data file

City**GRID**® data can be loaded from a file defined in XML format from UVM Systems GmbH:

1. The button `Load CityGRID Data` opens the pop-up window.
2. Choose *XML file* in the menu *Data Source*
3. Load the XML file by clicking the button ...

4.1.3 Define area of selection

For a simplified selection of bigger areas units can be combined as models. The model only works as a container for at least one unit but does not have any effect on the underlying data of the unit. One unit can belong to as many models as possible.

The area of selection can be defined via a model (recommended working procedure), and/or defined via a street and/or a rectangular window. This takes place in the selection window/pop-up window.

1. *Model*:

   activate checkbox *Model* and pick one of the models.

2. *Coordinates:*

   Activate checkbox Coord. and indicate left bottom corner as well as the expansion.

3. *Units/Street:*

   Activate checkbox *Units* and choose street. In order to limit choices, you can type in any part of the street name (in correct capitalization/use of small letters) into the line below

Note: The choice of the street is only possible in the database mode if a link to an address database is installed. In order to install such a link, please contact UVM Systems GmbH.

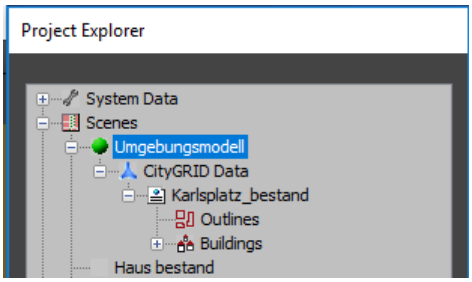
Note: The queries from all UnitIDs in the database can be very time consuming. Therefore, the selection on UnitIDs should only be carried out if necessary.

If more possibilities of choice are combined, the sum of the units is loaded.

Example: With the choice of model „City Centre“ and the choice of street „Main Road/Street“, that part of the main road can be defined as the area of selection, which is situated within the city center.

By clicking Load the window is closed and a new node with the model name is inserted below the node City**GRID**® Data in Project Explorer. The Selection Polygons of all units are loaded and a new entry in Project Explorer is added. The Selection Polygons (outlines) are only displayed in viewport if they are selected in the Project Explorer. They just serve to choose these units, which should actually be loaded.

Via the options window *Load CityGRID Data*, which appears below the Project Explorer, when the outlines are clicked in the Project Explorer, the choice of the City**GRID**® buildings to be loaded appears. If the Selection Polygons are selected here, the choice of the units to be loaded can be limited to those, whose selection polygons are selected in the windows (*Use only selected polygons*)



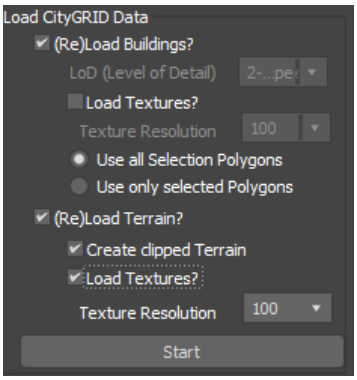
Note: the polygons for selecting the Units to be loaded, are only shown in the viewport when "Outlines" is selected in the Project Explorer node. For large models this can lead to (short) charging times.

4.1.4 Loading City GRID® Data

Once the *outlines* were clicked in the Project Explorer, the Properties window *Load CityGRID Data* appears below the Project Explorer. There, the City**GRID**® data to be loaded may be selected. A distinction is made between buildings and terrain models.

For loading buildings, proceed as follows:

1. Select checkbox at (*Re)Load Buildings*.
2. When working with a City**GRID**® database, via the checkbox LOD (Level of Detail), the complexity of the building may can be selected (see Manual City**GRID**® basics).



Note: selected objects are displayed in different colors depending on the status of the last valid version. If the colors have not been changed, the meaning is:

Light gray: The last valid version is a stable, non-torn version.

*Red: The last valid version was checked out in City**GRID**® Modeler (probably a different has already loaded it). If you load this Unit, the latest stable version is loaded.*

3. If the buildings have textures which are to be used, the checkbox at *Load Textures* hast to be set.
4. When working with a City**GRID**® database, the resolution of the loaded textures of a building may be set using the parameter *Texture Resolution*. The values of the pull-down menu represent the pixel size on the real object in millimeters. XML files automatically always use the highest resolution possible.
5. To restrict the set of the loaded structures, the option *Use only selected Polygones* can be set. Using this option, the polygones which have been selected in the viewport are relevant for which buildings are loaded.

Note:

*The units are stored in the City**GRID**® database in various levels of detail and can be loaded in each of these levels.*

The following LoD levels are implemented:

1 … block model

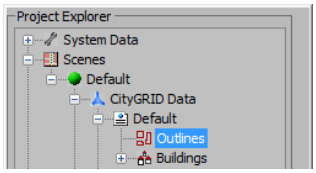
2 … form model (with roof shape and possibly also with roof overhang)

3 … detailed model (incl. dormers etc.).

To load terrain models, proceed as follows:

1. Activate checkbox *(Re)Load Terrain*.
2. The option *create clipped Terrain* loads only those triangles of digital terrain models, which are (at least partly) covered by the loaded buildings or by the loaded outlines. If the option is not set, all terrain triangles of the terrain model are loaded.
3. If the terrain models have textures, and these are to be used, the checkbox *Load Textures* must be set.
4. When working with City**GRID**® database, the resolution oft he loaded terrain texture can be specified by *Texture Resolution*. The values of the pull-down menu represent the pixel size on the ground. XML files automatically always use the highest resolution possible.

By clicking on Start copies of the City**GRID**® data are generated in the Builder project. This process may take several minutes for larger areas. During this time, a status window appears. Finally, in the project explorer, a node *Buildings* is created under the node of the loaded City**GRID**® model, under which the individual units can be found with their UnitID. Optionally, a node *Terrain* is formed.



4.2 Adding project data to the project

In general, project data is an external 3D model data created by architects, planners and 3D modellers. Furthermore, models can be created in the 3D-Studio.

4.2.1 Requirements for external data

To optimize the use of data created outside the City**GRID**® software, there are a number of constraints that must be taken into account in order to make it easy to incorporate the data into the Builder project.

- The geometry is to be created in the form of a triangle mesh, NURBS surfaces are not permitted.
- The buildings should be correctly placed in 3D. Subsequent georeferencing is usually extremely time-consuming and only partially accurate. The use of a blanket offset value in order to be able to obtain the required coordinate sharpness is permissible. If more than one building is being modeled in different files, always use the same offset value. The values of the offset must be given in the file name.
- The surface normals are invariably oriented in such a way that they always point away from the 3D object. This corresponds to a digitizing counterclockwise direction, which should be taken into account when making the model.
- The drawing unit must correspond to the unit of the other displayed data. In most cases, the unit will be meters.
- The geometry should reflect the characteristics of the building but be as economical as possible with the surfaces. The principle is: as few spaces as possible, as many as necessary.
- Building details that can be represented by texture shall not be modeled. In particular, fences, balcony railings, grids of all kinds must be made by means of texture.
- Interior of the buildings can be completely recessed. Navigating inside the building (e.g. in apartments) is not ideal with the Scout and should therefore be avoided. Large halls, on the other hand, can also be flown from the inside if the corresponding interior has been modeled. In particular, walls must be recreated for the interior with the surface normals towards the interior.
- The geometry must have triangular edges wherever different materials are to be applied. Each triangular area can carry exactly one material information.

- In addition to the building geometry, the geometry of the terrain model (planning model) must also be created in the immediate vicinity. The planned building models must be set exactly on the planning terrain model, so that high-quality visualizations arise and in particular a navigation from the pedestrian perspective is made possible.
- The material must be of type "Standard" (in 3D Studio Max). Any special materials that allow a realistic surface behavior (gloss, surface finish, etc.) can not be used in the real-time visualization, since the effects to be calculated can not be rendered fast enough with the available hardware. For a fluid movement 25, better 50 frames per second are necessary.
- From the default material, the builder evaluates only the diffuse color and the opacity channel. Thus, only these values may be defined, all other values are ignored when passing to the Scout.
- If a texture image is used on the diffuse color, it must be of the type "bitmap" and be of the type .jpg, .png or .tif. If an alpha channel is to be used, always use .png images. The Scout can take and apply the alpha channel directly from the png.
- Areas occupied by alpha channel texture images must be geometrically double in the model, with opposite normal orientation. Otherwise, the surface object will only be visible from one side. It is not sufficient to set the material on double-sided presentation; this information is not evaluated by the Scout! As an alternative to doubling the surfaces, it is also permissible to form very thin 3D bodies (such as cuboids).
- For each material using the texture image, the image scale must remain the same. This is defined by setting the so-called UVW coordinates (image coordinates). In most cases, it is possible to specify the original size of the texture images (eg modifier UVW Map in 3D Studio Max) and these values must be set the same for each surface textured by the material.

4.2.2 Importing external data

By clicking the button `Import External Data` in the main toolbar external data can be imported. The following formats are supported:

- AutoCAD *.dwg *or* *.dxf
- 3D Studio *.max *or* *.3ds
- VRML Virtual Reality Modelling Language *.wrl
- City**GRID**® Planner Projects *.cgp

After selecting a file the 3D-Studio dialogue for importing data appears (dialogue may vary, depending on file format).

Note: If available, de-select the checkbox „Reset Scene“ by any means so that present project data cannot get lost

By clicking `OK` the model will be implemented into the project as As-Built Data and can be easily moved into a Scene.

Note: Please mind that external data should exist in the same reduced coordinate system (project coordinate system), as it is used in the Planner Project. We recommend to transform the data into the project coordinate system in advance.

4.2.3 Import project data created in 3D Studio

Objects, which are created in the 3D Studio MAX, are implemented into the project as As-Built Data when updating the Scene Explorer (e.g. by clicking the button `Refresh view` in the main toolbar) and can easily be moved into a Scene: By doing so, the name of the object will be applied by 3D-Studio.

Advice: project data created externally or internally can be easily adapted concerning height via the button „Place Object on Terrain“ in the main toolbar (see III 6.1).

4.3 Data from Libraries

City**GRID**® 3D Studio Builder offers the use of libraries of 3D objects. Typically, those 3D objects are stored in a library that repeatedly occur in one or more builder projects. (for example, street furniture, vegetation, cars etc.)

4.3.1 Definition of a Library

The definition of libraries is up to the user. In 3D Studio Builder only the reference to the location of the 3D model itself is stored. Once created libraries are then available in every 3D Studio Builder.

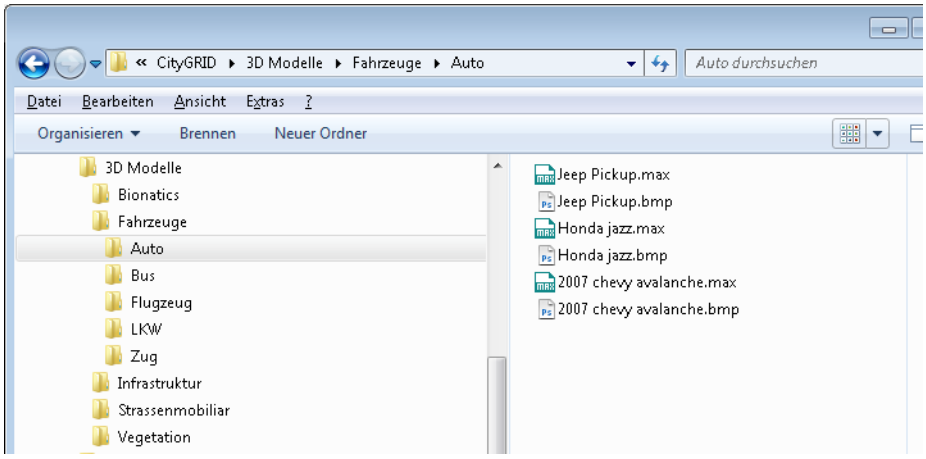
Libraries must have the following structure:

- For each library a separate directory has to be created. The directory name defines the name of the library. (for example, vehicles)
- Below that at least another directory has to be created. Each directory constitutes a seperate category of library. (for example, car, truck, rail, ship)
- This directory contains the actual 3D data which is stored in the .max format. Each .max file represents a library object type.

  <u>Note:</u> each .max file must meet the following criteria in order to be used as a library object:

  - *The geometry is present as polyface mesh (mesh).*
  - *Materials are defined with type standard and use texture images only in the form of diffuse color (Diffuse Color).*
  - *The origin of the coordinate system lies in X and Y in the center of the object and in Z at the bottom (at the lowest Vertex of the object)*
  - *The unit is identical to those of the other 3D data of the scene (typically meters).*

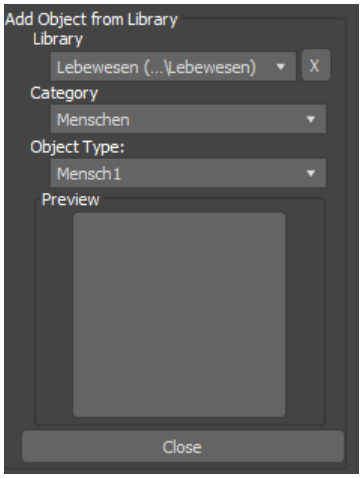
- At any .max file there may be a .bmp file of the same name, which shows a preview of the 3D object. These .bmp file must be created with a width of 95 pixels and a height of 85 pixels.



If the folder structure described above was created, you can make at any time an extension, modification or reduction of library object types. Any change to the .max files has an immediate effect on the 3D Studio Builder.

<u>Note:</u> Changes to the .max files do not affect already positioned library objects in scenes.

4.3.2 Creating a new Library

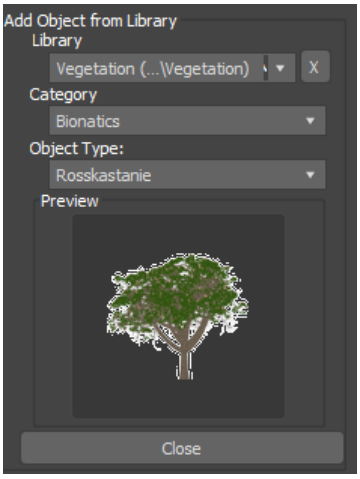


To include a new library, the following steps are necessary:

1. Start the Library Mode: click Button `Add Objects from Library` in the main tool bar. The window *Add Object from Library* is docked under the Project Explorer.
2. Select *Add library Folder* in the dropdown list of *Library*. Once this has been clicked, a file browser opens in which you can navigate to the folder containing the library to be included (see III 4.3.1.).
3. By clicking OK, this folder is included and the data in this folder are available as new library objects.

4.3.3 Adding library data to project

How to add geometrical objects from a library:



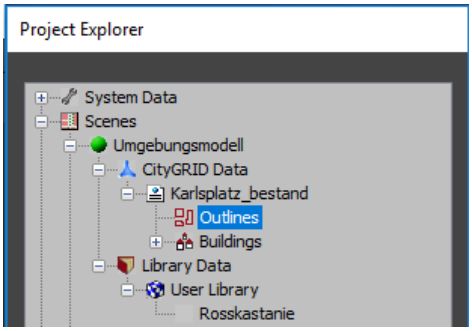
1. Start library mode: click the button `Add library objects` in the main toolbar. The window *Add Object from Library* will be docked under the Project Explorer.
2. Select *library*: You can choose between the libraries mentioned above.
3. Select *category*: library can consist of categorised objects.
4. Select object from library: Depending on the chosen library, ground plot and roof type, tree type, the category and/or the object type can be chosen
5. By clicking `Preview` the insert mode is started. As long as the insert mode is active, the pushbutton is marked yellow. Each input expected by the user in one of the graphics windows is indicated in the status bar.
6. Adding and positioning new object: The first click with the left mouse button in one of the graphics windows is always the 2D-positioning of the point of reference of the newly added library object. If the object was positioned within the existing area model, the objects is automatically put onto the area. Adding new objects can be cancelled by a right click in the graphics window.
7. Selecting the parameter: the second graphical indication mostly designates the rotation of the library object. There is the role for this parameter and all the other parameters (such as width, height, diameter, leaf density, etc.)
   - If the mouse pointer is within distance of the last mouse click, the standard value is used for the parameter. This position is indicated by a crosshair.
   - The further you move away (with the mouse pointer) from the position of the last mouse click, the bigger/smaller the figure of the parameter will be. The figure of the parameter used each time is shown in the status bar.
   - If you click the left mouse button, this figure will be applied and the selection of paramter skips to the next parameter.
   - If you click the right mouse button, this figure will be applied and the selection of parameter is cancelled (the remaining parameters will be set back to standard figures).

*Note: The pivotal point of building-library objects from City**GRID®** libraries is located in the left bottom corner.*

Advice: When inserting vegetation, it is recommended to switch to the high-quality plant illustration in the window in order to see the effects of the parameter settings.

8. Next object: after all parameters had been set or the parameter selection had been cancelled, the next object can be set with the next mouse click. Continue at point 5 or cancel the insertion of a new object with a right click.

9. Finalize inserting library data by clicking the button Close. The window *Add object from library* is closed and the Project Explorer is refreshed. The inserted library data is shown in the Project Explorer in the current Planning Variant (selected in the main window in the quick selection box below the node *Library Data*. By choosing the object in the Project Explorer its Options Window is docked, so that parameters can be adapted afterwards.



4.3.4 Set Library Objects with External Data File

In order to add a considerable amount of library data in one Step, a settings file can be prepared, in which essential information are quoted for each object, such as the library categroy, postition and scaling.

The settings file is an ASCII-file, in which a line is indicated for each object to be inserted. In each line parameters are separated by space or tabs.

The following parameters can be indicated:

- X, Y (mandatory): the position of the library object, for comma (,) period (.) has to be used!
- Z (optional): the height of the library object can be found out automatically.
- L, W, H (length, width, height) or H, D (height, diameter): measures in meters for the expansion of the object in the library.
- LN: Library Name
- LC: Library Category
- LO: Library Object

<u>Note:</u> If an object of a particular object type is not found in the specified category, no objects are placed. The hyphen (-) can be used as a wildcard to search all libraries and categories. If there is an identical name of libraries or classes, the first entry found is used.

We recommend to have this settings file in the reference coordinate system so that it can be used for various projects (see III 0)

<u>Example</u>: A settings file in format XYLWHKN

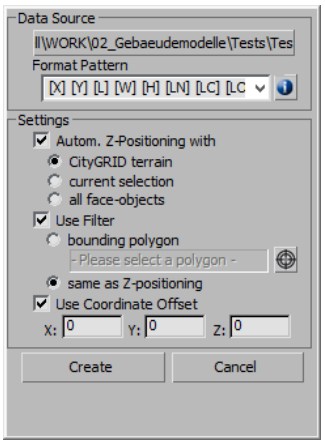
567898	*3678909*	*5*	*8*	*12*	*Street area*	*Lights*	*Lamp*
567887	*3678910*	*5*	*8*	*12*	*Street area*	*Lights*	*Lamp*
567895	*3678912*	*5*	*8*	*12*	*Street area*	*Lights*	*Lamp*

Three library objects „Lamp“ will be placed at the coordinates 567898/3678909; 567887/3678910; 567895/3678912. Each Lamp is 5m tall, has a width of 8 and a height of 5 meters. Find the library object „Lamp” in category “lights” of library “Street area”.

When adding library objects, you can:

- Automatically find out the height position of the objects.
- Limit the reasonable amount of library objects of the settings file via an area definition. Therefore, a single settings file of such a type can be used again for various projects, whereas only the objects, which are located in the corresponding project area, can be added to the project.
- Apply different coordinate offset to placed objects.

The automated insertion of the reasonable amount of library data works as follows:

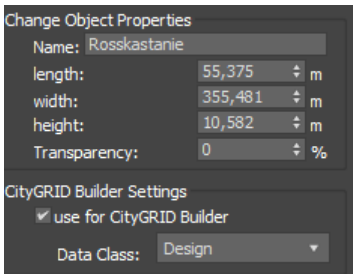


1. Click button `set library objects with external data file` in the main toolbar. Two windows are docked under the Scene Explorer:
   - one for showing the data source and
   - The other one for showing the settings.
2. Indication of the data source: by clicking the first push button, a browser opens in which a settings file can be selected. The formatting of this file has to be typed in as well. By clicking the `Info button`, a short description of the abbreviations is shown.
3. *Automatic Z-Positioning*: If activated, the Z-position of the inserted library object is traced automatically (and possible Z-coordinates of the settings file are ignored). The following methods can be chosen:
   - *CityGRID Terrain*: the object is set on City**GRID**® area models existing in the project.
   - *Current selection*: all the selected objects are used, whereas the highest calculated Z-coordinate is taken.
   - *All face-objects*: all objects existing in the scene are used, whereas the highest calculated Z-coordinate is taken.

<u>Advice:</u> Builder Control Center also allows the height positioning with help of the Config *page. (see Manual Builder Control Center)*

4. *Use filter*: Here you can restrict the area. Only objects, which are within the indicated polygons (which can be selected via the `Pick button`) or which can be calculated for a valid height are inserted.
5. *Use Coordinate Offset*: If the settings file contains coordinates in a different system than the reference coordinate system, the offset can be indicated for the reference coordinate system and not for the project coordinate system.
6. By clicking Create the library objects are added to the project.

4.3.5 Modifying library data



If you select a library object in the Scene Explorer, its parameter are shown in the options window. The features of the objects can be modified here and are accepted immediately (or after pressing enter)

Apart from the name of the object you can modify the basic settings, which are valid for all library objects (object length, -width, -height and transparency), as well as specific, object attributes dependent on the type of library object.

4.4 Deleting geometrical data from the project

Data can be deleted from the project via drag&drop in the Scene Explorer on Deleted Data.

<u>Note</u>: To permanently delete the contents of the Recycle Bin, the menu item "CityGRID Builder" of the 3D Studio Max main menu bar provides the function "Clear Deleted Data". By executing this menu item, the contents of the Recycle Bin are emptied in the Project Explorer and the associated backup files in the Builder project are deleted.

4.4.1 Restore Deleted Data

As long as the trash has not been deleted, data can be moved from this back into the project. For this, the data to be restored must be moved out of the trash on the respective main node (system data or scenes) using drag & drop.

Note: It should be noted that scenes can be deleted though, as described above, but can not be restored as a whole. When you delete a scene, all data are moved to the Trash, but the scene file itself will be destroyed. Therefore, to restore an entire scene with all the data you have to create a new scene or select an existing scene to receive the data.

5 *Add / alter / remove System Data*

System Data, as well as camera angle and Sun positioning settings are saved with the current project.

5.1 Camera

Cameras define viewpoints in the 3D scene. They can serve as a fixed starting point for exploring the Scout project, but also be used as an animated tour.

Each camera is defined by two points or between the spanned vector: The camera points and the target point. The camera point corresponds to the position in 3D space, on which the camera actually is. The target, however, is the point on which is the focus of the camera. The direction vector between camera and target point shall thereby determine the direction of view and the visible part of the 3D scene.

City**GRID**® Builder offers 4 types of cameras or positions :

- *Fixed Positions* :

  When using the fixed camera, eyes as well as target point are fixed.
- *Relative Position* :

  When using a relative camera, the line of sight is fixed. Based on the current target point of the window the viewpoint along the saved line of sight is set back.If you change to a geometrical data leaf in the Scene Explorer (a leaf is a node in the tree, which does not have a sub node), the virtual bowl is moved into the centre of gravity/mass of the new, active scene-node and a zoom on its extensions is additionally made.

  Relative cameras cannot be exported into a City**GRID**® Explorer project.
- *Circular Flight* :

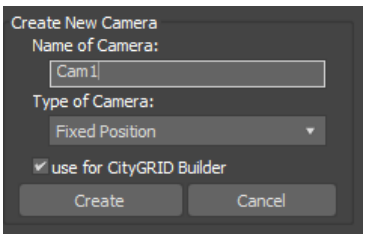
  At a circular flight, camera is moving, whereas the target point is fixed.
- *Camera Path Animation* :

  At an animation the path of the camera as well the path of the target point is moving.

When creating a new project, five pre-defined relative positions are set: the view from the four directions as well as the (topview) plan.

*Note: Camera animations and points of view can also be created directly in the City**GRID**® Scout (see City**GRID**® Scout manual)*

5.1.1 Create new camera



New camera can be created by clicking the node Cameras in Project Explorer. A window “Create New Camera” appears then in left lower corner:

1. Give the new camera a name.
2. Choose *a type of camera* (see III 5.1).

3. Click Create. The window *Create New Camera* is closed and the Scene Explorer is refreshed.

The newly created camera appears in the Scene Explorer under System Data – Cameras and is chosen automatically. By doing so, its Options Window is docked onto, so that additional parameters can be adjusted (see following section).

Depending on the camera type the new camera is created with the following settings:

- For fixed position the eye currently adjusted in the 3D window and the current target point are adapted.
- For relative position the current position adjusted in the 3D window is adapted.
- For a Circular Flight a circular camera path round the current target point (focus of perspective) set in the 3D window is set through the current camera. A selection can be made via "Zoom Extents Selected" by 3D Studio in order to focus the perspective.
- For a Camera Path Animation, a ¾-circle around the current target point adjusted in the 3D window (focus of perspective) is set through the current camera. Target point path identical to the eye point path.

5.1.2 Modify camera settings

If a camera is selected in the Project Explorer or via the quick selection in the main window, the 3D view is refreshed together with the camera angle.

If a camera is selected in the Project Explorer, the Options Window of the camera opens. Here the settings can be changed or further settings can be made:

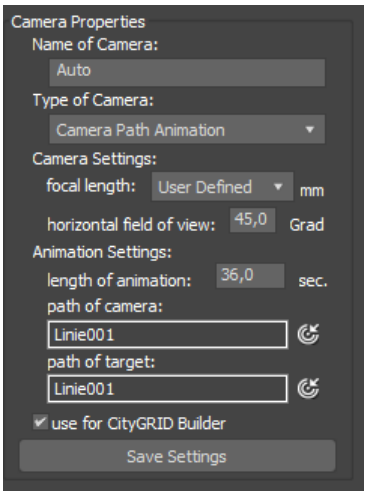
- For fixed and relative positions the view in the 3D window can be adapted and by clicking „Save Settings" the view can be saved.
- For circular flights and animations:
  - *Focal length*:

    50 mm corresponds to normal angle, 35 and 28 are focal distances for wide angle and super-wide angle, figures bigger than 50 are tele focal distances
  - *Horizontal field of view*:

    the apex angle which results from the focal distance adjusted above.
  - *Animation length*:

    duration of circular flight or animation.
  - *Path of Camera*:

    the path of camera is shown in the floor plan window and can be modified there. Via the `Pick button` on the right of the path name, any line can be chosen as a new camera path (the order to select the path appears in the status bar). A new path can be created via 3D Studio MAX functions (Create – Shape – Line).



<u>Note</u>: The procedure for creating new paths can be found in the 3D Studio Max Help. Newly created paths must not be included in the Builder project, the option "Use for CityGRID Builder" of the geometry object in the Project Explorer under Project Data must be removed!

  - *Path of Target*:

    For animation cameras a path of the target point can be indicated.

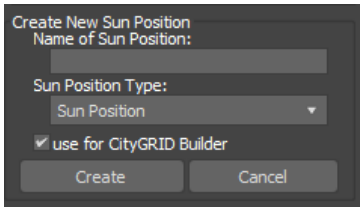
All the modifications are effective by clicking Save Settings.

5.2 Sun Positions

City**GRID**® Builder either supports statistic Sun Positions or dynamic Sun Course Animations. On the one hand silhouettes can be rendered and on the other hand shadow animations can be generated.

When creating a new project four pre-defined Sun Positions and 4 Sun Course Animations are prepared for each change of season.

5.2.1 Create new Sun Position

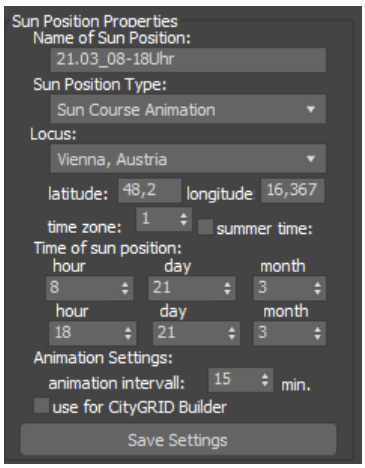


By clicking the node `Sun Positions` in the main window, the window for creating new Sun Positions is docked.

1. Assign name to Sun Position/Course Animation.
2. Choose the *Sun Position Type* (see III 5.2)
3. Click `Create`. The window *Create New Sun Position* is closed and the Scene Explorer is refreshed.

The new created Sun Position appears in the Scene Explorer under System Data – Sun Positions and is chosen automatically. By doing so, its Options Window is docked, so that additional parameters can be adjusted (see III 5.2.2).

5.2.2 Modify Sun Position Settings



If a Sun Position is selected in the Scene Explorer, also the Options Window of the Sun Positions is opened. Here settings can be changed or new settings can be made after creating a new one.

- *Locus*: By standard the indicated place is used when installing the software. The geographical latitude and longitude as well as the time zone of the place are adapted automatically if a different place is chosen from the list. Besides, this data can be adapted manually.
- Indicate if there is *summer time* at the point of time of the Sun Position.
- *Time of sun position*: Indicate date and time.
- For sun course animations: In the second time line, please indicate the end point of the sun course animation. Furthermore, the animation interval has to be typed in, the time interval in which the new shadow is calculated for an animation.

Modifications are applied when clicking Save Settings.

5.3 Remove System Data from the project

Data can be removed from the project via drag&drop in the Scene Explore on the Deleted Data except the five default cameras, such as north-, east-, south-, and west elevation and plan, as well as the Sun Position "switched off".

6 Tools

6.1 Place objects on Terrain

Geometrical data can be easily moved onto the terrain model in the Z-direction provided that a terrain model exists in the City**GRID**® data below the insertion point of the object(s).

1. Select the object(s) to be moved in the graphics window.
2. Click on the button `Place Objects on Terrain` of the main toolbar. For each of the objects to be selected the perpendicular/plumb line is set through the insertion point (centre for the object

coordinate system). Alongside these perpendiculars the deepest perpendicular is calculated. The deepest perpendicular for library objects from system libraries usually lies in the middle of the object but can also be located on the edge of other objects. The object is moved in that way that the deepest perpendicular lies on the City**GRID**® terrain model.

6.2 Duplicate Objects

Geometrical data can be easily duplicated. For example, if a tree is multiplied along a path, an avenue can easily be generated.

1. Select the object to be duplicated.
2. Click button `Duplicate Objects` in the main toolbar. (see III 2.1) An Options Window opens. Due to the current settings the positions of the objects to be newly added are indicated by blue crosses.
3. *Method*: Choose a method, after which the selected object should be duplicated. Available options are Orthogonal offset, Cycle offset and Path offset (see following sections)
4. Type: Choose the type, how the new objects should be inserted. This type offers the possibilities how objects in the 3D Studio can be administered.
   - *Seperate copies*:

     the new objects are generated as independent copies of the original.
   - *Instance Objects*:

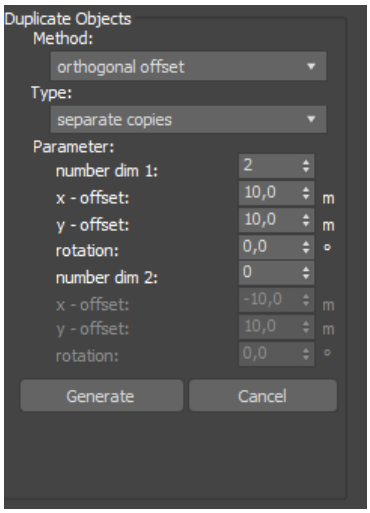
     An instance is an alterable copy of the original. Modifying the instance equals the modification of the original (recommendable, if all the objects should resemble each other)
   - *Reference Objects*:

     References are „One-way instances“. Modifications of the original object are also applied for the reference. Whereas modifications of the reference are not transfered onto the original (recommendable if each object is adapted individually, but all objects should get a different tree type at once)

   For a detailed description of instances and references we refer to the help section of the 3D Studio.
5. Click Generate in order to create new objects at the positions shown in the preview.

<u>Note:</u> For a detailed description of instances and references please refer to the help of 3D Studio.

6.2.1 Orthogonal offset



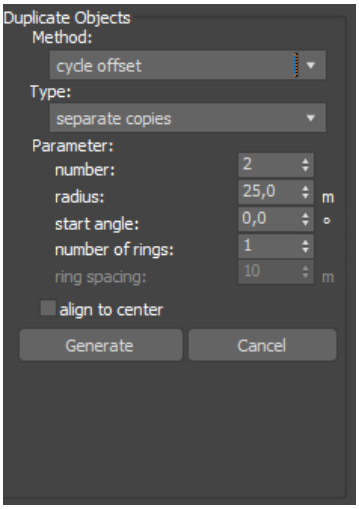
The new objects are generated in a regular grid/pattern.

- *Number Dim 1*: the amount of the objects to be newly created in the first dimension.
- *x/y-offset*: the direction of the first dimension.
- *Rotation*: the angle, around which each object is rotated anti-clockwise to its predecessor. For example, if an angle of 10° is indicated, the first newly created object is rotated by 10°, the second one by 20°, etc.
- *Number Dim 2*, *x/y-offset* and *rotation*: By the same procedure duplication into a second dimension can take place.
- Rotations from the first and second dimension are added.

<u>Note</u>: As shown in the figure 11 new objects are generated, because in the first dimension three new objects are created (makes four objects in this dimension) and in the second dimension two new objects are created (makes three objects in this dimension). In total there are twelve objects (11 new objects).

6.2.2 Cylce offset

The new objects are generated in a circle by the original object.



- *Number*: the number of the objects to be newly created on the circle.
- *Radius*: radius of the circle.
- *Start angle*: start angle of tangent of the circle through the position of the original object.

  Note: The figures of the angles mean:

  0: tangent parallel to the Y-direction, the circle is set left to the original.

  90: tangent parallel to X-direction, the circle is set below the original

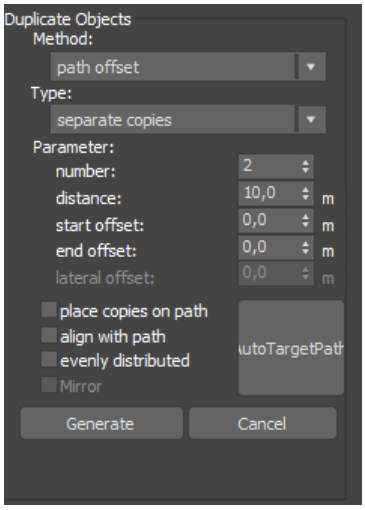
  180: Tangente parallel to Y-direction, the circle is set right to the original.

  270: Tangente parallel to X-direction, the circle is set above the original.

  In addition, any arbitrary angle is possible.
- *Number of rings, Ring spacing*: Starting from the first circle, additional concentric circles can be generated, whereas the radius is respectively magnified by the distance between two rings. The number of created objects refers to the number per circle when there are several circles.
- *Align to center*: Objects can be rotated that way that all objects show the same side to the central point.

6.2.3 Path offset

The new objects are generated alongside the path. First, click the button `Select Path` and select a path in one of the graphics windows.



- *Number, Distance*: the number and the distance of the objects to be newly created along the path.
- *Start offset*: the distance of the first object to be created from the starting point of the path (measured along the path)
- *End offset*: the distance of the last object to be created from the end point of the path (measured along the path); can only be chosen if „evenly distributed“ is active.
- *Lateral offset*: positive or negative normal distance of the objects to be created from the selected path; can only be chosen if „place copies on path“ is active.
- *Place copies on path*: indicates if the objects to be newly created are generated on the path itself or on a parallel of the path through the position of the original object.
- *Align with path*: indicates if the objects to be newly created should be rotated together with the direction of the tangent of the path.
- *Evenly distributed*: indicates if the objects to be newly created should be evenly distributed along the path. In this case a start and end offset can be indicated, but not the distance (since it is generated automatically).
- *Mirror*: if there is a lateral movement/shifting, the objects can be doubled by additionally generated mirrored round the path (example: generation of a doubled tree row of an avenue with a street axis as path)

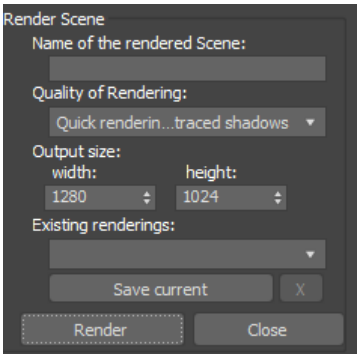
  Example: Form an alley by using a street axis as path and parameter mirror enabled.

6.3 Refresh view

By clicking the button `Refresh View` in the main toolbar, on the one hand, all the windows are drawn again and on the other hand the Scene Explorer is refreshed.

7 *Render Animations and Images*

By clicking the button `Render Animations and Images` in the main toolbar the Options Window is docked onto the Project Explorer. For the rendering current camera angles and the Sun Positions-settings activated in the quick selection are used.



1. Select the Viewport, from which you want to render the video/image.
2. Type in a *name* of the rendered Scene Animation/view.
3. Choose a *quality* of rendering for the shadows (the higher the quality, the longer the rendering time)
4. Define the *Output Size* of the rendered animation/view.
5. By clicking `Render` the render window of Autodesk 3D Studio MAX, in which you can see the image or in which the rendered frames can be seen, opens. You can cancel the rendering process by clicking the "Esc" button (the result which has been rendered up to that time, remains).
6. If a video is rendered, a Video Player appears in order to view the video.
7. Repeat Steps 3 to 6 that often until the result meets your expectations.
8. Press `Save current` in order to save the rendered result. The animation/view is created in the project directory in the subdirectory "Media". Furthermore, this animation/view is adapted to the list of existing renderings, so that they can be quickly re-loaded later on. By clicking "X" the current rendering is deleted.

Note: If a circular flight or a camera path animation as well as a sun course animation is active, the length of the rendered animation is determined after the Sun Course Animation. In order to guarantee that within this time the camera covers the whole animation path, the length of animation at the camera angles must be set on the same time, like the one which results from the Sun Course Animation (Time span and interval).

*Note: The creation of a virtual flight through the 3D scene via Camera Path Animation mostly offers the most impressive views. However, the definition of the path as well as the creation of the animation requires a lot of experience working with Autodesk 3D Studio MAX. The City**GRID®** Builder only offers the possibility of the choice of path for the camera and target point. Modifications of the path design as well as the animation procedure have to be defined autonomously in Autodesk 3D Studio MAX and do not belong to tasks of City**GRID®** Builder.*

*Note: Camera animations and points of view can also be created directly in the City**GRID®** Scout (see City**GRID®** Scout manual)*

For further information see Autodesk documentation.

8 *Export to City*GRID® *Scout*

8.1 Control of Exports to City GRID® Scout

For each entry, except for the four main nodes, it can be specified in the project explorer whether the data should be transferred to City**GRID®** Scout (This applies to the selected node and the underlying nodes). For this, the checkbox `Use for CityGRID Builder` must be set. If this parameter has not been set, the appropriate data remain in 3D Studio Builder, but are not exported.

*Advice: By selectively setting the parameter "Use for CityGRID Builder" construction aids can be used in 3D Studio, without requiring them to be deleted before exporting the data for City**GRID**® Scout.*

For geometrical data, the data class can be defined as well. Using the data class controls the degree to which the relevant data group is to be optimized for the Scout. At your disposal are the following classes:

- Terrain

  covers all 2.5D data representing terrain or surface models. Large models should be divided into a regular grid in order to allow a better optimization. If textures are used, a steady mosaic of images should be used (ideally all images or orthophotos should have the same dimension.).

- Buildings

  All semantically correct City**GRID**® buildings fall into this class. Unlike other 3D data objects of these data class are structered in the roof and façade elements (or any other elements) and have relatively compact dimensions. When deriving LoDs, the building is geometrically simplified more and more and, in extreme cases, can also be completely removed (eg. small objects at a great distance).

- Design

  All 3D data to be assigned to neither the terrain nor the building data class can be found in this class. Unlike the other two classes of data, data of type design can not use the the special optimization algorithm for the generation of LOD levels, which is why the performance of the scene can be affected in very large scenes. This is in turn ensures that all data, even in low LOD levels, are always present and never be removed.

Note In general, the data class is set automatically by 3D Studio Builder and need not be changed.

8.2 Create City GRID® Builder Project

The City**GRID**® **3D Studio Builder** is an ideal tool for preparing scenes for the City**GRID**® Scout. After the scene had been finished in Builder, it can be exported into a Builder project. The Scout adapts Scenes, positions, etc. from the Builder project if these are activated in the corresponding Options Window for the integration in the City**GRID**® Scout and allows the interactive navigation through the scene.

The export is started by clicking the button `Transfer Data into Builder Project` in the main toolbar. The Builder project is created in the subdirectory "Scout" of the project directory. The City**GRID**® Scout settings of all objects are taken into consideration (see III 8.1). In this process a number of optimizations for the City**GRID**® Scout are executed. Depending on data volume the process lasts a couple of minutes. At the end a notification appears that the process has been finished.

Note: If the SHIFT key is pressed when you click on `Transfer data into Builder project`, Builder Control Center opens immediately after raw data processing.

`Transfer data into Builder project` + ALT key opens the Scout project, if this was ever processed via Builder Control Center.

IV. Builder Control Center

Builder Control Center is the central management tool for Builder projects. Using Builder Control Center new projects Builder can be created, reset or deleted. Other tasks of this program are the compilation of scenes to variants, as well as combining multiple Builder projects. Using Builder Control Center data optimization for City**GRID**® Scout projects can be started, which is the essential step in creating the project from the pre-processed raw data. Finally, Builder Control Center allows unlocking of already created Scout projects for distribution.

Note: Builder Control Center is currently only available in English.

1 Types of Builder Projects

Using the Builder Control Center allows you to create and manage different types of builder projects. There are three types of Builder projects that can be defined.

1.1 Simple Builder project

The simple Builder project is created by the raw data processing in 3D Studio Max or FME. In a simple Builder project the raw data are actually physically present. The raw data can be created in several cycles by FME and / or 3D Studio Max and be stored in the form of scenes.

During the optimization for City**GRID**® Scout, Builder accesses the raw data and builds the data structure of the Scout scene. The Scout project can then be unlocked and distributed. (see IV 5.3)

Simple Builder projects are suitable for local projects, or for processing of data from a particular data class (buildings, grounds, point clouds, etc.). They are the basis for merged builder projects as described below.

1.2 Managed-Builder project

Managed builders are associations of simple builder projects. They reference the included simple builder projects and access the raw data stored in them. The raw data itself remains with the included simple builder projects. During the creation of the Scout (Create, see 5.1), data trees are created from the raw data for each file class, which prevent subsequent separation of the raw data. The result is an optimized data management, however, every change of a referenced subproject requires the complete recalculation of the Scout. Just as with simple builder projects, a variants can be implemented.

Note: Managed builders can not be added to any other Managed Builder projects because they no longer have raw data. However, these are absolutely necessary when optimizing the Scout project.

To create a Managed-Builder project, an empty Builder must be createdy (see III 3) in which the desired Builder projects are included (see III 4.1).

1.3 Merged Builder Projects

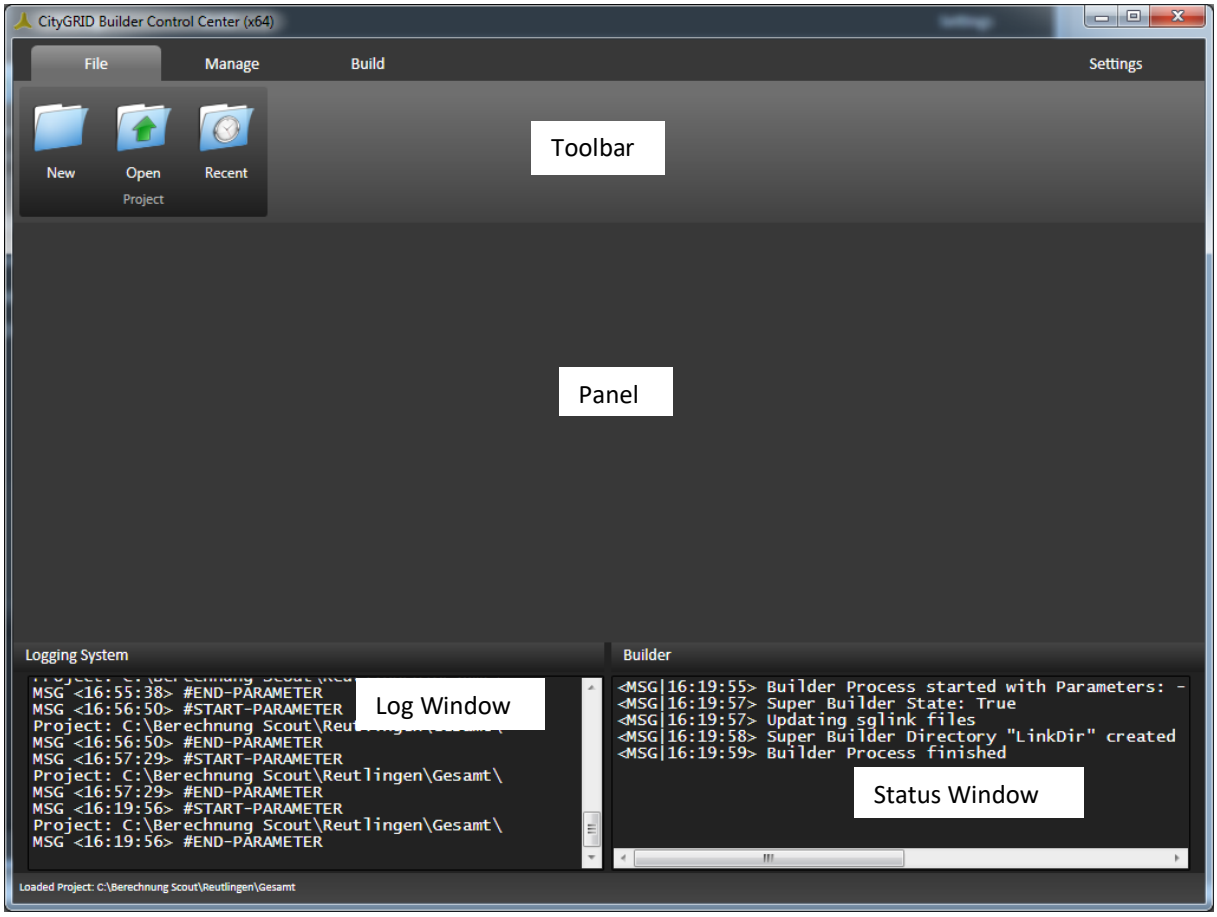
Merged Builder projects reference simple or Managed-Builder-projects and combine Scout-projects within them. There is no optimization of the Scout-projects anymore, but only the joint representation of the sub Scouts. There is no access to the individual data in this step. Each sub Scout appears with its set configuration and variants, which can still be influenced via the *Config* page (see IV 5.3).

As long as a merged scout has not been published, the scout accesses the integrated partial scouts locally. Only when you publish all partial scouts are copied into the scout directory of the merged builder project.

Note: merged Builder projects can be added to any other Super-Builder or merged Builder projects.

To create a merged Builder project, a new Builder project must be created (see IV 3). Then the individual Builder projects are included. (see IV 4.1)

2 The Builder Control Center Interface



The surface of the Builder Control Center is divided into the following areas:

- *Toolbar*

  Here, the program functions are located. Using the tab system different functions can be accessed.

- *Panel*

  Depending on the selected toolbar function, the panel is used for additional settings or for starting.

- *Log Window*

  Each started process logs in the file of SuGu.log of the Builder project. This log file is forwarded in real time to the log window.

  <u>Note:</u> For deleting the log file, navigate to the tab Settings and click on `Reset` button.

- *Status Window*

  The status window shows start and end times of the current Builder processes.

3 Tab File

In File tab new Builder projects can be created, or existing ones can be opened.

By clicking on the button `New` a new Builder project is created. This opens a file browser, where the Builder folder structure (and therefore the builder project) is set. Through this process, all other functions of Builder Control Center become active.

To open an existing project, you have to click on `Open`, and select the Builder Project from the File Browser. There is a file builder.sgp (see I 2), which can be opened with City**GRID**® Builder Control Center. If this file is selected, the Builder Project in the Control Center opens for all functions.

Alternatively, an already published Scout Project (see. IV 5.4) can be opened. Change file type to "Scout project (sgp)" and to choose the project.sgp or projectStarter.sgp file of Scout project.

<u>Note:</u> Opened Scout projects can only be re-published (see. 5.4) all other functions of the Builder Control Center are not available. A renewed publishing of Scout projects is necessary if an updated version of the Scout software is to be linked to the data or Scouts settings are subject to any change.

When starting the Builder Control Center, the list of recently opened projects (Builder and Scout) will appear in the panel. Use button `Recent` to reaccess this list at every time. A project can be started by clicking on the `Open` button or by double-clicking. `Remove` eliminates the selected entry from the list of recently opened projects.

4 Tab Manage

In *Manage* tab settings can be made for the administration and combination of builder projects.

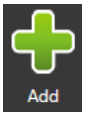
4.1 Projects

For the creation of Scout scenes, the data to be visualized must be organized and held in Builder projects. If the projects do not become too large, a simple Builder project is mostly sufficient. For very large projects, it is better, however, to distribute the data packets in different builder projects and create the Scout scene by combining several Builder projects.

<u>Advice:</u> The distribution to several Builder projects can be done either by the data class (buildings, terrain, external data), or by administrative units (map tiles, district boundaries, etc.).

Click on the button `Projects` for activating the mode for combining (linking) existing Builder projects.

Note: If Builder projects are to be combined, you must at first create a new, empty Builder project by clicking the button New *at the* File *tab. With help of* Project *button, this Builder project can then be used for linking the already prepared Builder projects.*

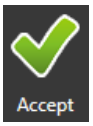
By clicking on the button `Add` 

a file browser opens in which an existing Builder project can be integrated. For this purpose, choose file builder.sgb in the Builder project. For each builder project that has been selected in this way, a line appears in the panel with the path to the file builder.sgb. In this way any number of projects Builder can be integrated.

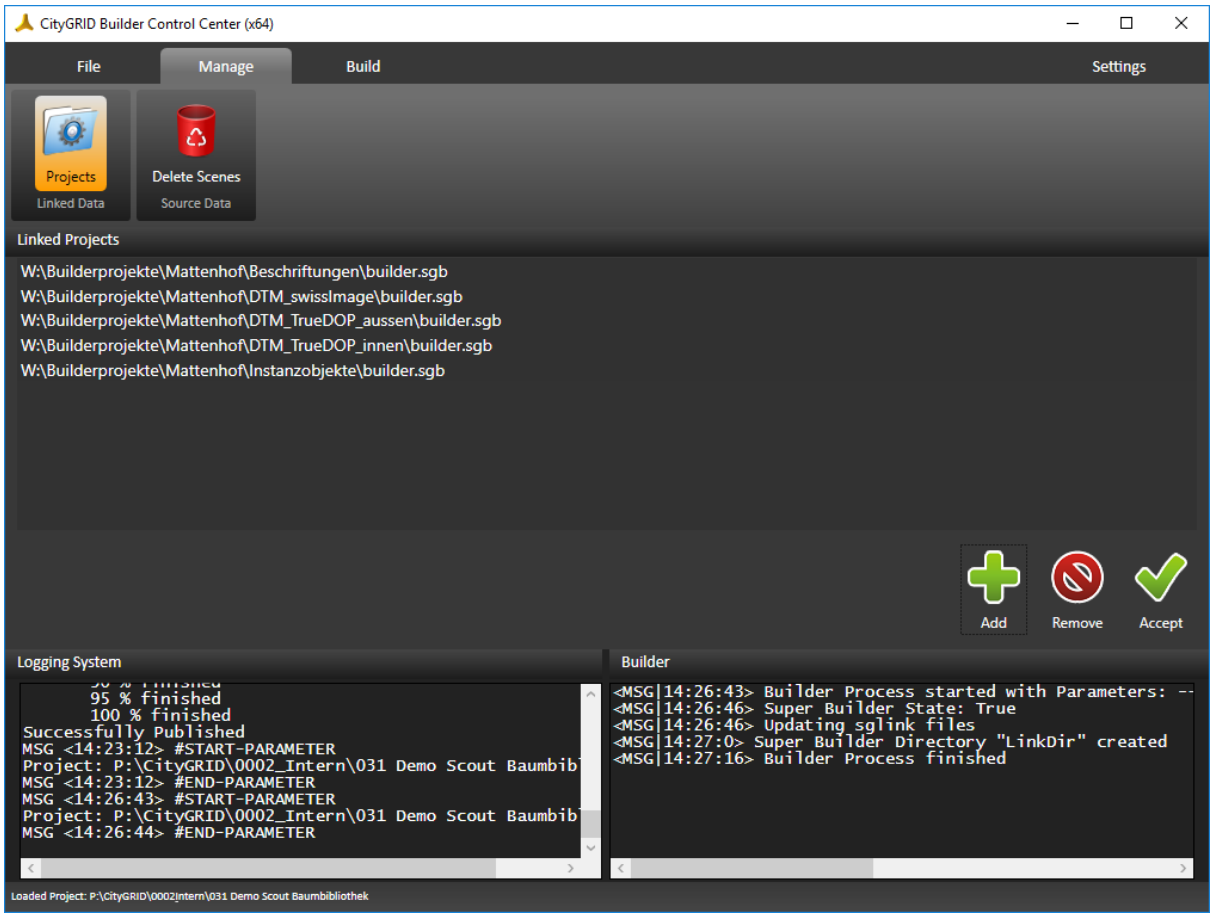
<u>Note:</u> Meged Builder projects can not be included with help of the button Projects*. A merged Builder project represents the final stage of a Builder project and can´t be processed further.*

Unwanted Builder projects can be selected in Builder Control Center panel and can be removed by clicking the button `Remove`. 

After completion of work, the selection made is confirmed with button `Accept`.



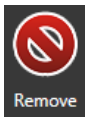
<u>Advice:</u> If you switch the tab before the changes have been confirmed, Builder Control Center will send a message to the user. Use this message to discard unwanted changes.

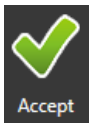


4.2 Delete Scenes

By clicking the button `Delete Scenes` you can access the scenes of the read in Builder project. Use this function to delete unneeded data from the Builder project.

The control panel displays the scenes grouped by the data classes. By marking the checkbox, which is located behind the path, the corresponding scene can be marked. As an option, the checkbox that is located at the end of the line of data class, may be selected or unselected. That way, all scenes of the data class may be selected or unselected.

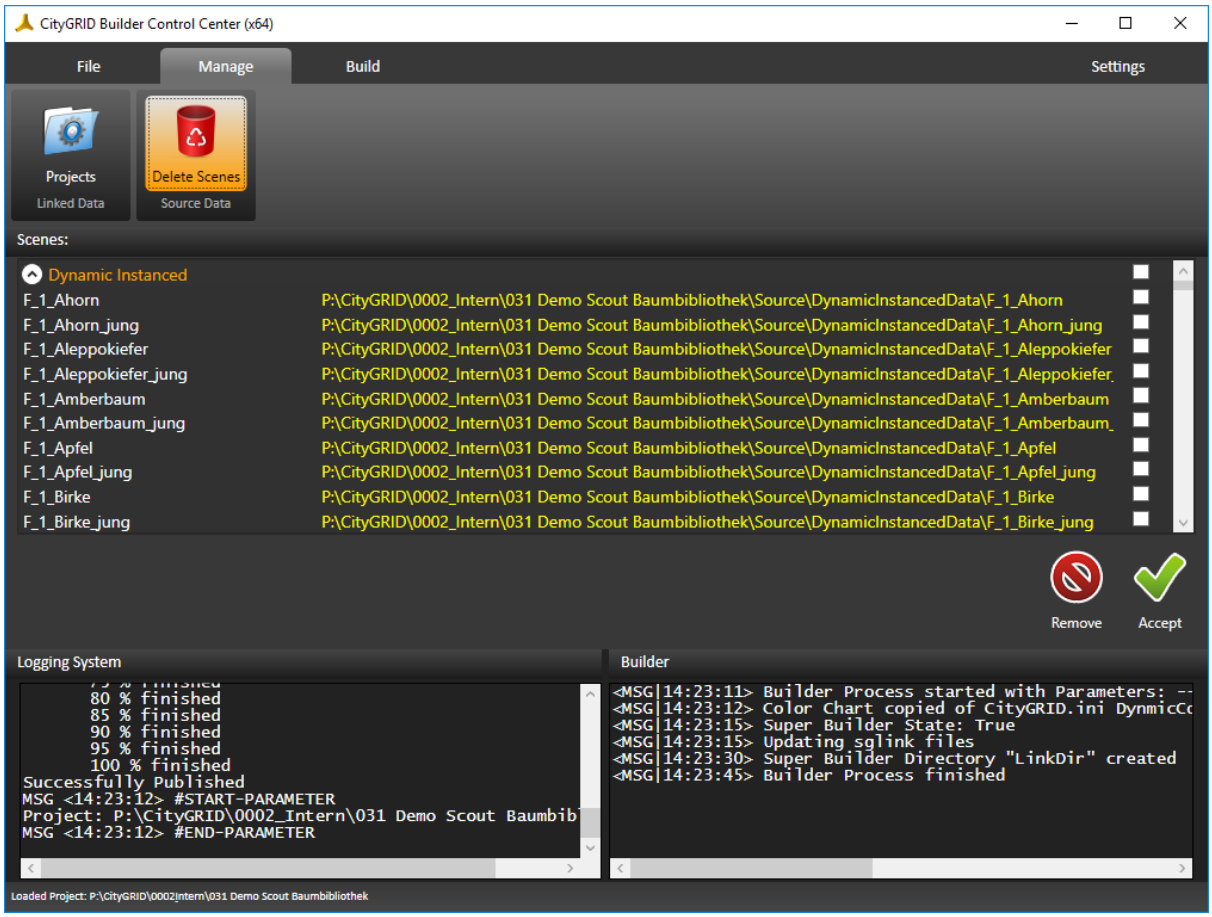




By clicking on the button `Remove` all selected scenes to erase are marked and deleted from the control panel, but the data itself is not changed yet.

Only when you click on the Button `Accept` Builder Control Center will delete the marked files in the Builder project.

<u>Note:</u> This action can not be undone. The deleted data will be physically removed from disk. When you click on Accept *no other confirmation prompt appears.*



5 *Tab Build*

In this tab, the processing of the Scout project is started and the distribution outside the licensing system is prepared.

5.1 Optimization for CityGRID® Scout

All scene data from FME and 3D Studio Builder must be optimized and restructured prior to loading in City**GRID**® Scout. In this process, the previous files are resolved and combined to variants. In addition, LoD levels of geometry and images are created, data trees (voxel octree) are formed and new files which are optimized for performance will be generated. The control of these operations on the `Create` button

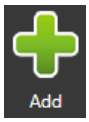
5.1.1 Take over Scenes and Variants in CityGRID® Scout

The concept of variants makes it possible to show scenes in different combinations in City**GRID**® Scout and switch at runtime. Each Scene of a Builder project can be assigned to any number of variants and a Builder project itself may have any number of variants.
Variants only control the visibility of scenes while the Scout project runs. The data itself is just calculated once and sent to the graphics card only when needed.

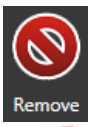
In the panel all the scenes of the loaded Builder project appear, these scenes have been generated by the FME Builder or with the 3D Studio Builder. The scenes are grouped by data classes (see II 2) and the source-Builder project is specified (in yellow) next to the scene name.

Note: In the case of Super-Builder all the scenes of all Builder projects involved appear.

If variants have already been defined, these are also read and Builder Control Center shows which scenes are assigned to which variants. If no variants exist in a project, Builder Control Center automatically creates a variant "New Variant".



New versions can be created by clicking the `Add` button. Any name can be assigned by Double-click on the variant name in the left column of the control panel.



Not needed variants can be removed from the Builder project using `Remove` Button.

Note: Variants can also be compiled from the processed data packages via the Config page and do not have to be defined during Create. To do this, each data packet that is to be incorporated into a variant must be created individually. The recommended procedure is to define variants when combining builder projects in a merged builder..

5.1.2 Process Modes

Depending on the size of the Scout project, there are two alternative optimization methods: *Project* and *City*. These two methods are offered in the *Properties* Window. The two methods process the Scout data in different ways and have different demands on used software modules and computing times. Also, the data handling in the Scout is fundamentally different.

Note: Both modes use the GPU for rendering the texture images. For this reason, it must be avoided that the GPU becomes inactive during the calculation, or not available. In particular, these effects occur when screen savers activate or when working on the Remote Desktop Connection of Windows. Therefore, pay attention to the Windows settings that the graphics card is never turned off and use third-party software (such as TeamViewer) for remote control of the computer.

Process Mode Project

The processing mode *Project* is intended for local, small-scale projects. All data is processed so that they can be completely held in the main memory or graphics memory. Applying processing mode *Project*, there will be no reloading of data. The textures are processed in different resolutions (LOD) and the geometry files generated are optimized for the Scout. At the start of the project Scout determines the amount of available storage and automatically selects the possible resolution of the textures. The geometry is always displayed with the full amount of data.

Note: Due to the ease of data processing, the computation time is significantly shorter than in the processing mode City. Nevertheless, computation times of several hours may occur even in this mode.

Process Mode City

The process mode *City* is availabe for the visualization of large-scale projects (entire cities or regions) with high data density. In contrast to *Project* mode, Scout now applies a distance-dependent visualisation of the geometry and texture data. Scouts loads the data dynamically at runtime, so at close range Scout always visualizes the best possible resolution of geometry and textures. The further away data are from the current focus of Scout project, the more generalized they are displayed. As a prerequisite, City**GRID**® Builder organises the geometry data in the form of data trees, which store the data in its original resolution and additionally in generalized Levels of Detail (LoDs). 3D Studio Max functions are used for generating the LoDs. For that reason, a license of 3D Studio Max has to be available at the computer. If no 3D Studio Max is available, just a simplified tree structure can be calculated without the LoD stages. Due to this, range of visibility will be limited when Scout visualizes this simplified structure.

Note: High computation times and considerable volume of data can be expected when applying this method of processing. Computing might take several hours or even days.

5.1.3 Settings

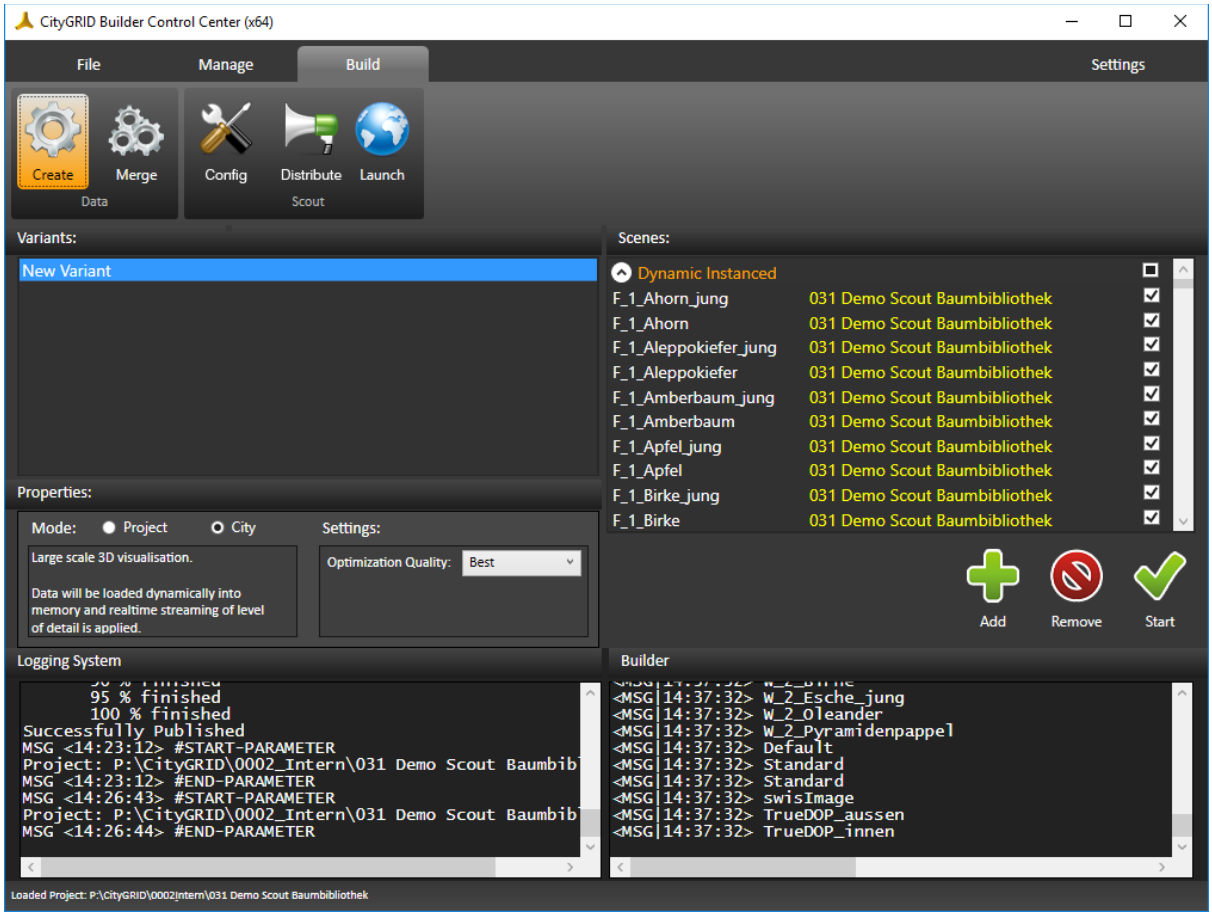
When optimizing the Scout data textures have to be transferred to an optimized data structure. Depending on the quality of the data used, this optimization can be applied more or less what has immediate effect on the calculation time and the quality of representation in the Scout.

In addition to optimization quality integration of any City**GRID**® attribute data can be made.

Optimization Quality

This value determines how well the optimization of the data trees may, in particular as regards the texture optimization, take place. Decisive for this is the sensible allocation of textures to surfaces. When textures are placed on several areas of different extent and orientation pixel resolution cannot be calculated correctly. The consequence is that the textures in the Scout appear coarse or greatly faded. In these cases, the *optimization quality* settings must be gradually reduced to achieve visually pleasing results in Scout.

*Advice: Data textured by City**GRID**® data may be processed in best optimization. For external data, it depends on the type of preparation, if sacrificing optimization results can be archived. Foreign data should therefore be conducted in their own Scout projects, which can be subjected a new optimization if necessary.*



5.1.4 Carrying Out the Optimization

To assign scenes a specific variant, you have to proceed as follows:

1. Create a new variant or activate an existing one.
2. In column *Scenes* you have to set the checkbox of assigned scenes.

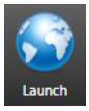
   Advice: If all scenes of a data class (Static Building, Dynamic Design, etc.) are to be allocated, the checkbox next to the class name can be activated. If all scenes within a data class are set, a square is drawn into the checkbox instead of the hook.

3. Repeat process for all variants.

4. Set the desired processing mode in *Properties* Window.

5. Initiate optimization of the Scout project by clicking `Start` 

Note: Once `Start` is clicked, the changes are carried out and stored with the Builder project. Ongoing optimizations can't be terminated early.



The end result of the successful optimization of the data is a Scout project. The button `Launch` becomes active in this case, as well as button `Distribute`. By clicking on `Launch` the just calculated project is opened and visualized by City**GRID**® Scout, provided that an appropriate license is available on the computer.

Note: If the creation of the Scout project is not successful due to a problem, the corresponding log file with an exact description of the fault can be found in the log window. Please send them to support@uvmsystems.com together with a description of the genesis of the builder project.

5.2 Merging of calculated Scouts

Large Scout projects can be created efficiently, from previously formed partial Scouts which are brought together for display. That approach is followed by the merged Builder or merged Scout. Previously optimized Scouts are assigned to a new Builder project and combined on the creation method `Merge`. The advantage of this approach is that each partial Scout can be optimized and inspected for itself. Only when these subprojects were created to satisfy, the unification takes place.

Advice: If variants are to be displayed in a merged scout, simple builders or managed builder projects with calculated scouts must be created and combined via the Config page (see Config5.3).

A merged Builder project currently represents the final state of a Builder project. Such Builder projects can not be combined any further.

5.2.1 Creation of a merged Scout

1. On tab *File* click button `New` 

to form a new Builder Project.

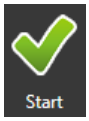
2. Switch to tab *Manage* and hit `Projects` 

to assign simple or Super-Builder projects with already optimized Scouts (see IV 5.1).

3. Swicht to tab *Build* and press button `Merge` 

to perform combination of previously linked Builder projects.

   Note: If one or more of the linked Builder projects has no valid Scout, an error message appears and the unification is not performed.

4. Click `Start` 

to begin unification. In file ProjectStarter.sgp the Scout folder path is entered for each linked Scout project. On the data of the linked Scouts itself no change is made.

   Note: The merging of Scouts is done in a few moments because extensive optimization work no longer has to be carried out. However, the Scout can only be started if the referenced data can also be accessed. To distribute such a Scout, the referenced Scout data must be copied into

the scout directory of the merged Builder project, which is possible via the distribute page (see 5.4).

5. Use Button `Launch` 

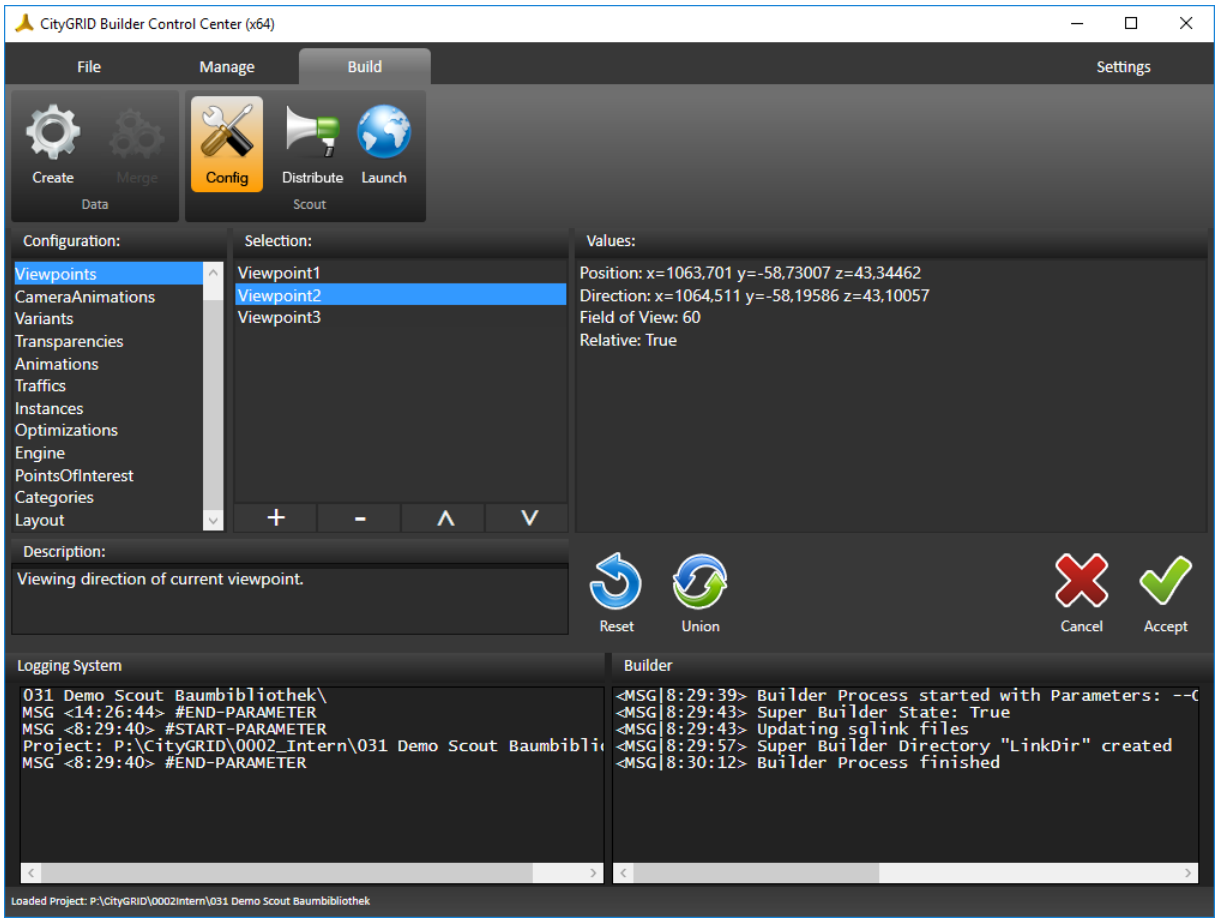
to start merged Scout.

Note: Since no new data arise during merging merged Scouts are usable only on creating computer. Only by publishing the Scouts copying of the components takes place in the Builder project (see IV 5.3).

5.3 Config

The button `Config` allows access to various settings that determine the final appearance of the Scout. For example, viewpoints, variants or general startup settings of the Scout can be defined on this page.

The *Config* page is composed of the lists *Configuration*, *Selection* and *Values*, as well as the text field *Description* with description text for the respective parameter and the button bar. The mode of operation is the same for each parameter. In the *Configuration* column, select the category for which you want to set parameters. As soon as a choice has been made for Configuration, (pre-) defined parameter sets appear under *Selection*. A choice under *Selection* finally displays the parameters to be determined in the *Values* field.



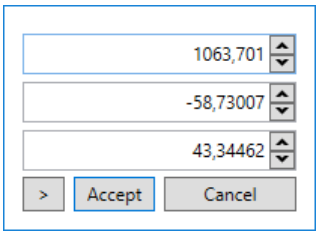
Note: Not every entry under Configuration *has parameter sets listed under* Selection*. Some categories directly open the parameters under* Values*.*

5.3.1 Set parameters

The setting of parameters via *Config* always follows the same scheme:

1. Under *Configuration*, select the desired category. The available parameters then appear in the *Selection* list, or the buttons for creating / editing are activated.
2. Select the desired parameter set under *Selection* to get the editable values in field *Values*. If no parameter sets can be assigned to the selection under *Configuration*, you can start editing the parameters directly under *Values*.

   However, if the choice allows you to define parameter sets, you can use the button + to create new ones or - to remove existing ones. The order of the parameter sets can be changed via the buttons ∧ ∨.
3. Under *Values* now the corresponding parameters are to be set. These can either be attribute values or data packages to specify variants, transparencies etc. Each entry in the list *Values* can be edited by double-clicking. Depending on the type of parameter, different inputs are possible. The range extends from simple text entry, through the selection of predefined lists, to the transfer of values directly from a running City**GRID**® Scout.



   The acquisition of values from a running Scout takes place by means of the button > and is intended in particular for taking over positions and viewing directions.

   Note: In order for the Builder Control Center to be able to communicate with the Scout, the "Communication" option in the Settings → Application menu must be activated in the Scout.

   Note: For entries with decimal places, the comma of the current keyboard layout must always be used as separator.

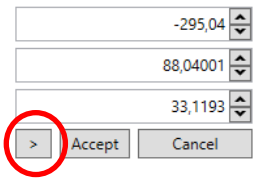
   Each value is assigned a descriptive text in the text field *Description* that provides information about the function or the expected input.

5.3.2 Available Parameters

- *Geodetics*:
  - in order to be able to use an overview map, it is necessary to enter an *EPSG code* (http://www.epsg-registry.org/) of the data in Scout. The numbers of the code must be given.
  - *Offline Map* controls via the values true and false whether the Scout project should be searched for data for an offline overview map. If such data is found, the map window uses it instead of the set Online web map service. It should be noted that the use of the offline map still requires an upright internet connection in order to be able to display the placement of the current position in the 3D project in the overview map.
  - if a local coordinate system is used that is not assigned to an EPSG code, the offset values can be specified under Custom Offset, which must be added to the local coordinates in order to get the EPSG coordinates.
  - For shadows, the position of the sun's position in the Scout must be determined in geographic coordinates for Longitude and Latitude. A maximum of three decimal places are taken into account.
- *Viewpoints*:

  Is the list of all viewpoints in all (sub) Scouts of the (merged) Scout. The names of the viewpoints can be changed in the *Selection* list by double-clicking. With a double click on the parameter values of a selected viewpoint, these can also be changed (Position, Direction, Field of View).

  Tipp: If a corresponding Scout project is open and external communication is allowed (see Scout manual Chap.5.4.4), coordinates of the current camera position can be taken over with the > button in the dialog opened by double-clicking on the values.



- *Camera animation*:

  Shows all animated flights of the Scout. The names of the animations can be changed in the *Selection* list by double-clicking.

  *Advice: Animations come from 3D Studio Max or are created directly in the Scout (see manual City**GRID**® Scout)*

- *Variants*:
- The category *Variants* displays all variants of all (sub-) Scouts as well as the underlying data packages and offers the possibility to influence the variant switching after optimization.

  Advice: Subsequent variant definition is necessary for unified (merged) Scouts (see IV 5.2).

  In the *Selection* list, existing variants can be renamed by double-clicking. The button + creates a new variant, - removes the selected variant.

  The *Values* list displays all data packages, grouped according to the embedded Scouts. By setting checkboxes, they can be combined to new variants or existing ones can be resolved.

  Note: A data packet was created when optimizing a scout (see IV 5.1) and represents the sum of all scenes that were assigned to a variant when Create *was carried out.*

  The name of the associated builder project is given for the assignment of a Sub-Scout.

  Advice: Ideally, only data of one topic and one data class are included for each Sub-Scout. This means that the name of the Builder project clearly shows which data package is assigned to a variant).

  If a data package has not been explicitly assigned to a variant, it appears in the *Values* list with the text *not specifically assigned* and the data type *Root* or *Scene*. Otherwise, the name of the respective variant of the Scout, and its data type (Instance, Animation, Traffic, etc.) is listed. In order to assign a data package to a variant, you must set the corresponding checkbox.

- *Transparencies*:

  Like *Variants*, offers the possibility to switch individual data packages transparently. Transparent areas can be faded in and out during the runtime of a Scout (see City**GRID**® Scout manual)

  Note: 'Transparencies' has no effect on materials with an alpha channel. The transparency values of the textures remain unchanged

- *Animation*:

  List the definition of all moving objects of a (sub) Scout. The animations are defined in 3D Studio Max.

  Note: Please contact UVM Systems to implement animations in the Scout.

- *Traffics:*

  List all traffic objects of the (Sub) Scout. Currently not implemented.

- *Instances*:

  Allows height positioning of instance data. Similar to *Variants*, all data packages of (sub-) Scouts are listed under *Values*, which can be used for the desired 3D positioning. By setting a checkbox, the data package is linked to the instance objects. By clicking on the `Move to` Button, each instance object determines the highest intersection of all data packages for its 2D position and places itself on it.

Note: Requirement for the height positioning of instance objects is either a published Scout (cf. 5.4.3) or a Super-Builder (cf. 1.2) in which instance data and surface data are held together.

- *Optimizations*:

  Allows setting the visibility distance in the Scout. Valid values are -10 to 10. These values correspond to the preset distance levels, which can also be set via the Scouts Settings → Performance menu. The lower the value, the faster the scene loads. In return, the view is further limited accordingly.

  Note: The desired values can be determined with the Scout running via the menu → Settings → Performance. It should be noted that increased visibility places significantly greater demands on the computer, so it must be ensured that the defined values remain processable. If the performance limit is exceeded, corresponding entries can be found in the Scout's SuGu.log.

- *Engine*:

  Represents all the general parameters of the Unity 3D Engine, such as background color, double-sided rendering, inactivity countdown, dot size in point clouds, and so on.

- *PointsOfInterest*:

  After one or more categories have been created under Categories, POIs can be created here. Under Values of a selected POI the following values are essential for the creation:

  - GlobalPosition: Position of the POI popup, can be chosen between 10 values. Default values are sufficient.
  - GlobalDuration: Duration of the POI popup display in seconds.
  - Global Offset: Here the position of the POI popup can be changed, relative to the specified position in pixels.
  - Category: Here the parent category of the POI is set.
  - Position: Here the position of the POI is set. If external communication has been allowed in the currently opened, associated scout project and the current camera position corresponds to the desired POI position, the scout position can be adopted with the > button in the coordinates dialog. The POI position is the center of a bounding box.
  - Visible Delta: Here the dimensions of the bounding box are specified (in meters) in which the POI is to be displayed in the scout. Outside this bounding box, the POI will not be displayed.
  - Is Relative: Usually these are relative coordinates in scout projects but can be changed here.
  - Viewpoint Position: Here the coordinates of a viewpoint are specified, which was ideally created in advance and named the same as the corresponding POI. This viewpoint is the view that is jumped to when the POI is selected in Scout.
  - Viewpoint Direction: This is the direction of the POI viewpoint.
  - Viewpoint Field of View: This specifies the angle of view of the POI viewpoint in degrees.
  - Viewpoint Relative: Are the coordinates of the viewpoint relative?

  When the settings have been accepted with Accept, the opened Scout must be closed and reopened for the changes to take effect.

  A meaningful image can be transferred to the POI popup. In the file system of the Scout project under ScoutName/Content/Data/Images/POIs images can be created in .jpg format, which must have the same name as the POI itself. Recommended are image sizes not larger than 1024*640 px.

- *Categories*:

  Enables you to set Points of Interest categories.

- *Layout*:

  Defines logos, or the caption of the Scout.

- *Project*:

Allows you to set parameters that apply to the entire Sub Scout

- *Texture mode*

  offers the possibility to influence the type of texture tiling. The Repeat, Clamp, Mirror and Mirror Once methods are available. Repeat instructs the 3D engine to apply the texture repeatedly. This setting applies to data from the 3D Studio Builder that was created with tiled textures. Clamp maps the texture image once on the 3D surface. This mode is to be used for terrain models that have been textured using orthophotos, as well as for CityGRID® buildings with automatic texturing. Mirror mirrors the textures on one axis. This texture mode is only to be used in exceptional cases.

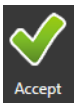
- *Cull mode*

  defines the visibility of surfaces in the 3D window. By default, only those surfaces are visible in 3D windows whose surface normals are oriented against the direction of view of the viewer. The system, off, front and back options are available. The system obtains the settings from the "Double side" parameter of the Configuration Engine, "Off" always shows all surfaces on both sides, "Front" fixes the visibility on surfaces whose normal points against the direction of view, "Back" in turn always only shows surfaces whose surface normals are facing has.

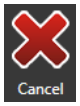
Note With the cull modes off, front and back, the setting Doubleside has no more influence and during the runtime of the scout it can no longer be changed using the F7 key..

`Reset` discards all settings and reads the settings from all sub Scouts. All settings that were set purely in the Builder Control Center will be lost by this action.

`Union` combines all the current settings with those of the sub Scouts This measure may cause entries such as viewpoints to be duplicated because they do not match existing entries. If necessary, unwanted entries must be removed manually.



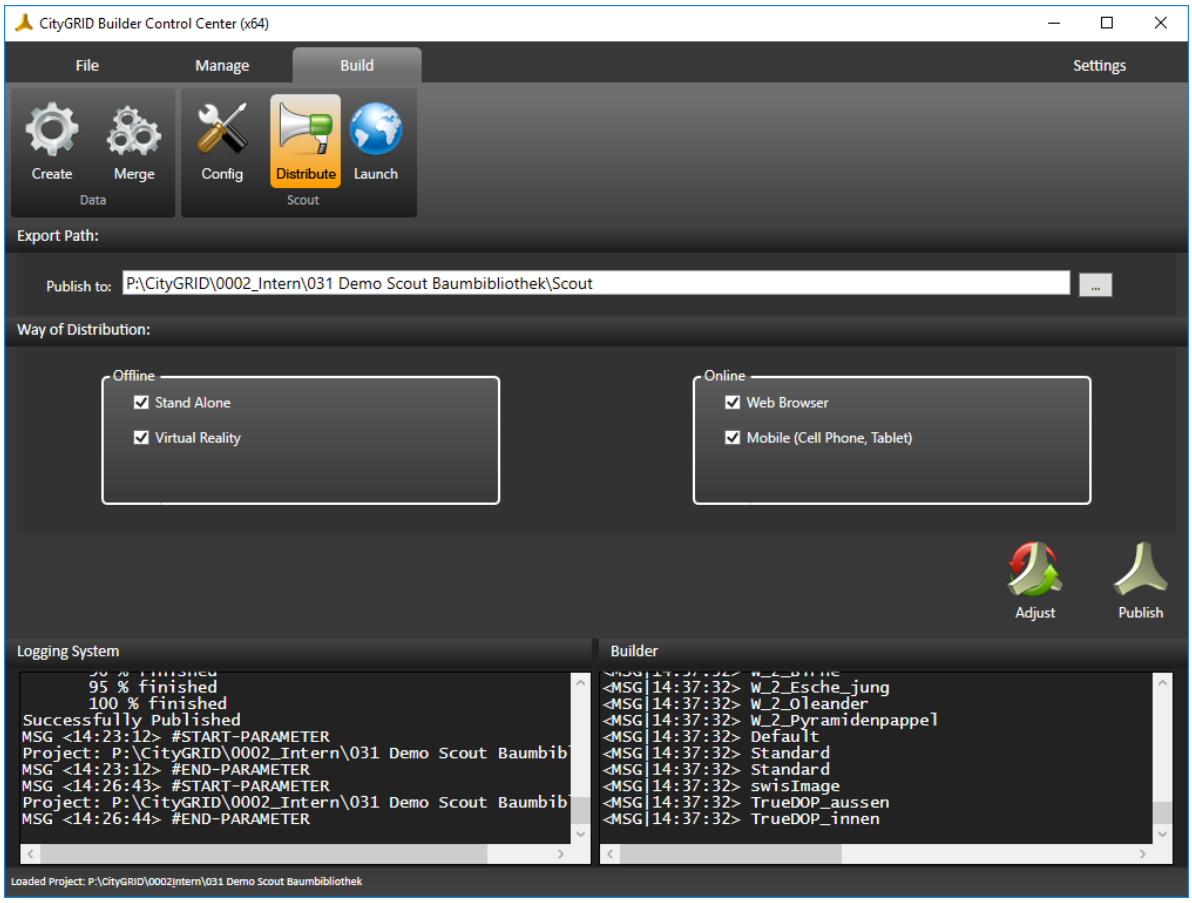
`Accept` takes over all changes and stores them immediately in Scout. This applies equally to both previously published and unpublished scouts. The *Config* page is closed.



`Cancel` discards all changes and closes the *Config* page without changing the Scout.

5.4 Distribute

Properly calculated Scout projects are displayed by default with the installed version of the City**GRID**® Scout. This version requires a connection to City**GRID**® license stick and is therefore unsuitable for the distribution of the 3D scene. In order to make the Scout project accessible to a wider circle of users, a publication must be made. Through this step, the Scout receives all necessary data viewer data to be able to run independently of any City**GRID**® installation.



Note: Prerequisite for a pulication is a corresponding license. For further information, please contact UVM Systems GmbH.

5.4.1 Export Path

When publishing a Scout project, the Scout data in a Builder project must be prepared for final processing. For example, a copy of sub Scouts may be necessary, or the preparation of the image material for online distribution. In any case, the Scout directory of the Builder project fills with the final data. The *Publish to* parameter can be used to specify the location of this process. The button opens a file browser in which the location is specified. The processed data can be picked up at this location and further processed according to the distribution route.

Note: This step does not change the data of the Builder project, the project remains valid after publishing and can also be changed.

5.4.2 Way of Distribution

The publish ist available for two distribution channels:

- Offline
- Online

In the case of offline distribution, the entire Scout project is copied to a client computer and started there via Scout.exe. The transmission can be done by means of a suitable data medium (DVD, USB stick, FTP server, etc.).

- *Stand Alone*: Creates an interactive 3D VR scene based on the Unity 3D environment (https://unity3d.com/unity). The Scout project is an independently executable program that can be started on any modern Windows PC without further installation. In addition to the free navigation in the scene, for example, shadow animations, attribute queries and measurements in 3D space can take place without any further precalculations.

- *Virtual Reality:*

  Prepares the scout project for display in VR glasses such as Oculus Rift and HTC Vive.

The online distribution uses a client-server architecture to bring the Scout data to the user. The Scout project is hosted on a web server (Apache or IIS) and sent to the client via an internet connection. The data is visualized on the client computer via a standard web browser. The software independently recognizes the currently displayed area and loads exactly the required data (streaming). For the transfer to mobile devices (mobile phones, tablets) is a separate app available, which can be obtained through the official app stores (Google Play or Apple App Store).

Note: When transferring the Scout data considerable amounts of data, which lead quickly to the utilization of data volume, will occur, especially when used on mobile devices. Usage within a wireless network (WLAN) is therefore strongly advised.

- *Web Browser*:

  Prepares the scout data for display in popular web browsers (Google Chrome, Mozilla Firefox, Microsoft Internet Explorer, etc.). The display in the browser takes place using WebGL (preferred solution) or Java, depending on which technology the browser supports.

- *Mobile (Cell Phone, Tablet)*:

  Prepares the Scout data for use in its own Scout app that can be used on mobile devices. The apps are available for all popular mobile operating systems (Android, iOS, Windows Mobile) and can be obtained from the respective app stores. The linking of the Scout project to the app is controlled by the server that hosts the data.

Note: To carry out this type of distribution of your Scout projects, please contact UVM Systems GmbH. You will then receive support in setting up the web server and embedding your Scout projects.

5.4.3 Publish a Scout

1. A Builder project with calculated Scout (see IV 5.1) must be read into the Builder Control Center.
2. Select the `Distribute` button on the *Build* tab.
3. The button indicates the location of the Scout directory
4. The publication is started via the `Publish` button. Depending on the selected distribution path and its sub-settings, the Scout data is processed. Depending on whether a simple copy process or a complex reorganization of the scout is necessary, the time required for this work step varies.

Note: Before publishing, it is recommended to have already configured the Scout via Config (see 5.3). Although a Scout can still be adjusted after publishing, the adaptation will only remain in the Builder project and thus for re-publishing if the Scout is still in the Builder project. If the Scout was published to another location, there is no further connection to the Builder project.

5. To synchronize local changes with online Scouts without republishing the data, click the `Adjust` button. This also applies to the synchronization of settings that have been set via the *Config* page (see IV 5.3)

Note: Published scouts may only be modified later to a limited extent. If too much changes are made to the data, a new publication of the project is necessary.

6 Settings

Settings lists functions for the operation of the Builder Control Center, as well as the settings for the use of multi core support.

6.1 Multi Core

To speed up the optimization of Builder data for the Scout, the optimization process (see IV 5.1) can be set to work in parallel mode. This principle is called multi-thread processing. By activating the check box *Use multicore threads* the parallel processing is activated. The selection list *Threading quality* now determines the degree of parallelization

- *Low:*

  Determines the number of CPU cores and allows exactly one work process per thread.

- *Medium:*

  Per CPU core, the maximum number of CPU cores allowed as threads (for example, 4 cores allow 4 * 4 = 16 threads), but never more than a quarter of the maximum allowable by the operating system.

- *Full:*

  The number of threads will be determined by the operating system and will take full advantage of the computer's full computer capacity.

Note: Multi-threaded should only be used when no other programs are running on the computer, as the resource consumption by the Builder, and the sub-processes it controls, can be very large. Especially at Full *the builder consumes the entire system resources! For this reason, it is advisable to outsource the optimization process to an independent workstation. If no own computer is available, Multi Thread should only be used with the* Low *setting.*

6.2 Manual

The `manual` button opens the builder manual. The Builder Control Center continues to run in the background and can be operated in parallel to the manual.

Note: To display the manuals, a pdf reader must be available.

6.3 Info

The `Info` button opens a window with general information about the City**GRID**® installation. In case of support please add a screenshot of this window to the mail with the error description.

6.4 Reset

With the `Reset` button all log files of the loaded Builder project are emptied. Please reset all logs before a misbehavior is reproduced and reported to UVM Systems.

V. Error Reporting

The City**GRID**® software is developed, tested and used internally by UVM Systems, and strive for an error-free product. Nevertheless, we cannot guarantee that errors will not occur while your work with it.

Through experience we are aware that the following errors may occur:

- Autodesk 3D Studio Max crashes without prior warning/message: The reasons for such fatal errors are usually problems of 3DS itself and are beyond our control.
- If you get a City**GRID**® Error messages it is typically because of corrupt data. Usually, continuing to work is possible.
- Script-Errors: The reasons for these are often mistakes in operating with the software. In most cases continuing working is possible. If further problems occur, we recommend exiting the Plug-in City**GRID**® and starting it again.

If you want to send an error report, please enclose the following information/data:

1. Verbal description of the actions which produced the error
2. Log file: You can find the protocol in your profile under: \AppData\Roaming\\CityGRID\CityGRID.log. Find the log files also in the specified place if you've set the Log Path in Settings menu of *City**GRID***® Administrator
3. Every Builder projects stores its log, called SuGu.log file in Builder project folder.
4. Scout log, also called SuGu.log, can be found in Builder project folder under Scout/Content/Data/SuGu

Note: Please backup this file immediately after a crash, because the renewed starting of the software may cause the file to be overwritten.

Advice: In 3D Studio Max, the directory of the log file can be opened using the menu „CGBuilder – Logging".

Please send error reports to support@uvmsystems.com

VI. Contact

UVM Systems GmbH

www.citygrid.at

www.uvmsystems.com