

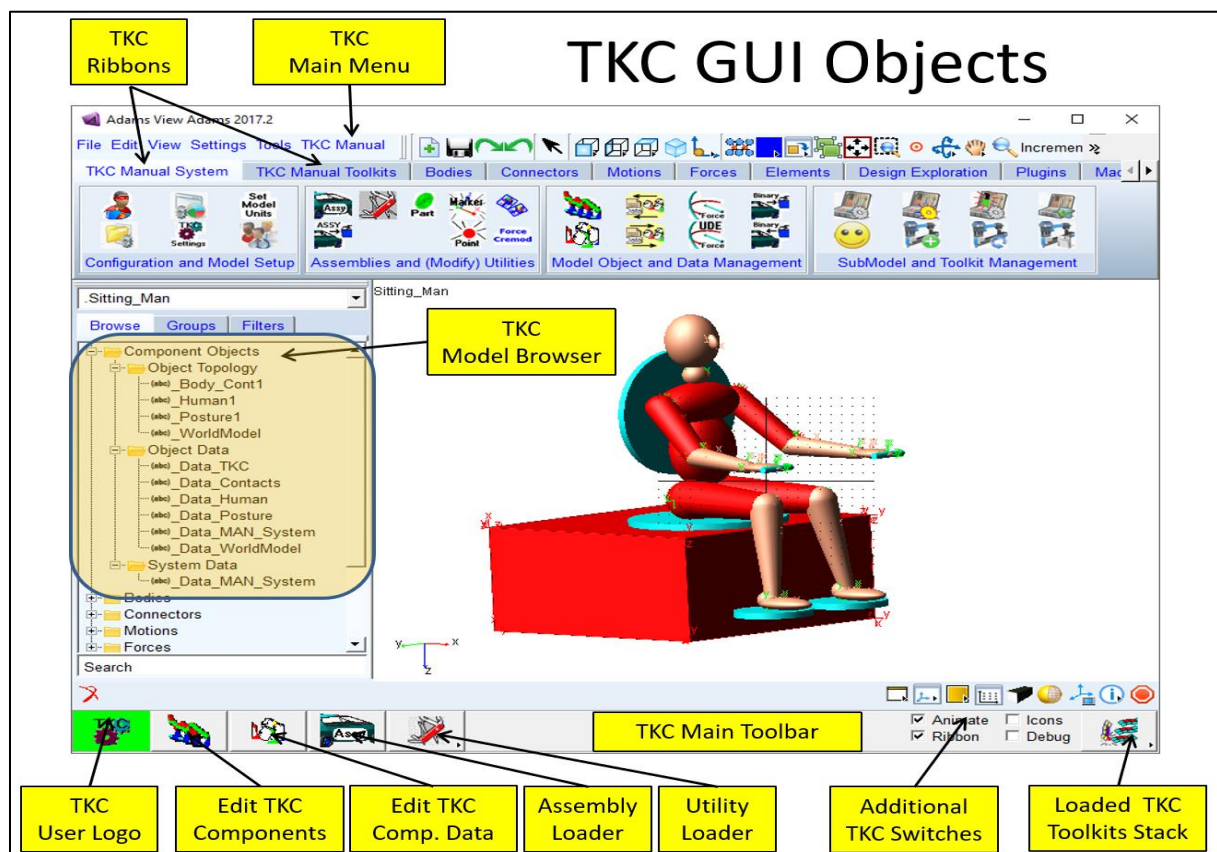
Menus and Generic Dialogs

Table of Contents

- 1: Introduction
- 2: TKC Main Menu
- 3: TKC Control Dialog
- 4: Main TKC Menu Macro Functions
- 5: Model Object Popup Menus
- 6: Measure Display Popup Menu
- 7: Generic Toolkits Dialogs
- 8: Component Object Manager Dialog
- 9: Component Data Manager Dialog
- 10: Utility Loader Dialog

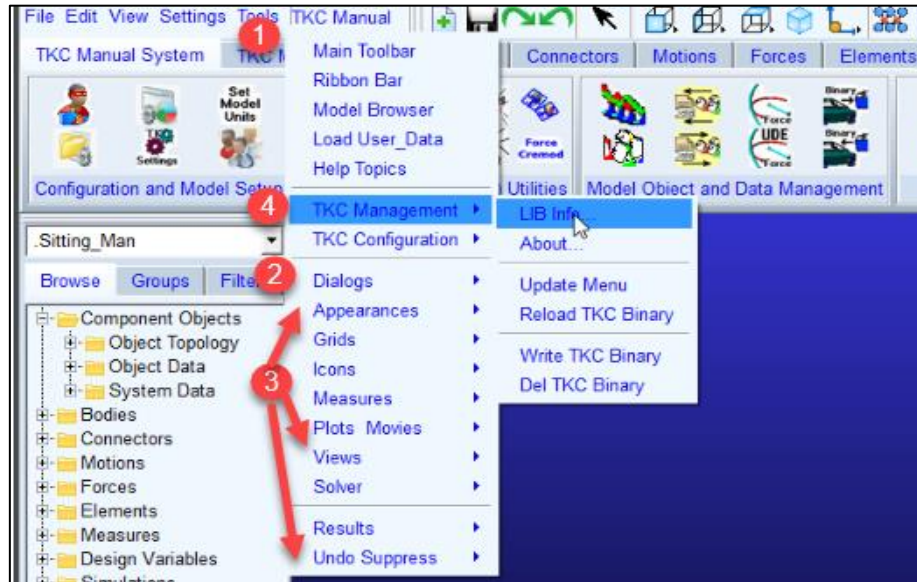
1: Introduction

In this section, the different menus and generic dialogs of TKC will be explained and illustrated. After this section, you will know the difference between standard ADAMS/View and the added TKC functionality and have an impression of the specific User interaction elements of TKC and how they can be used to efficiently build and manage models and model components.



2: TKC Main Menu

The TKC Main Menu is displayed by (1) pressing the *TKC_Project_Name* label shown on the ADAMS/View Main Menu. The TKC Main Menu contains (2) a selection of buttons to display specific TKC dialogs or (3) to perform typical macro actions that increase your efficiency and speed up repetitive modelling and simulation tasks.



Some typical management tools (4) are stored in the TKC Main Menu to update the Menu, verify TKC session settings and manage the TKC binary file contents.

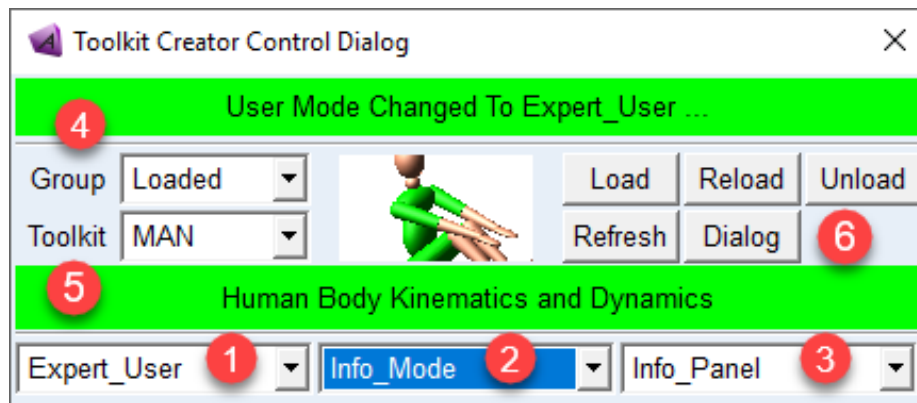
3: TKC Control Dialog

The TKC *Control Dialog* is opened from the *TKC Configuration* group in the TKC Main Menu. This dialog is used for advanced user control actions and management of TKC toolkits connected to ADAMS.

It allows users to switch from *Standard_User* to *Expert_User*. In expert user mode, more functions to manage and debug TKC Toolkits are activated. All Toolkits are defined as ASCII macro and data files, in expert user mode, TKC automatically checks for updates in code ASCII files and converts them into the binary TKC Toolkit data. At the cost of a slight speed decrease of macro processing, this drastically increases the efficiency of debugging new macro functionality. Thus, in expert user mode, deleting and re-creating components and component data triggers an update of the macro from the respective ASCII macro files.

Two other list options are available in the Control Dialog for activating additional info ((2): *Info_Mode* or *Verbose*) and for redirecting additional textual information to a reserved Dialog ((3): *Info_Panel* or *Info_Prompt*).

The upper part of the Control Dialog is reserved for managing the Toolkits loaded in TKC. The left side of the dialog contains two dynamic option lists, the upper list (4) allows selecting the Toolkits *Group* and the lower (5) allows selecting a *Toolkit* in the selected Group. Available *Group* options are:



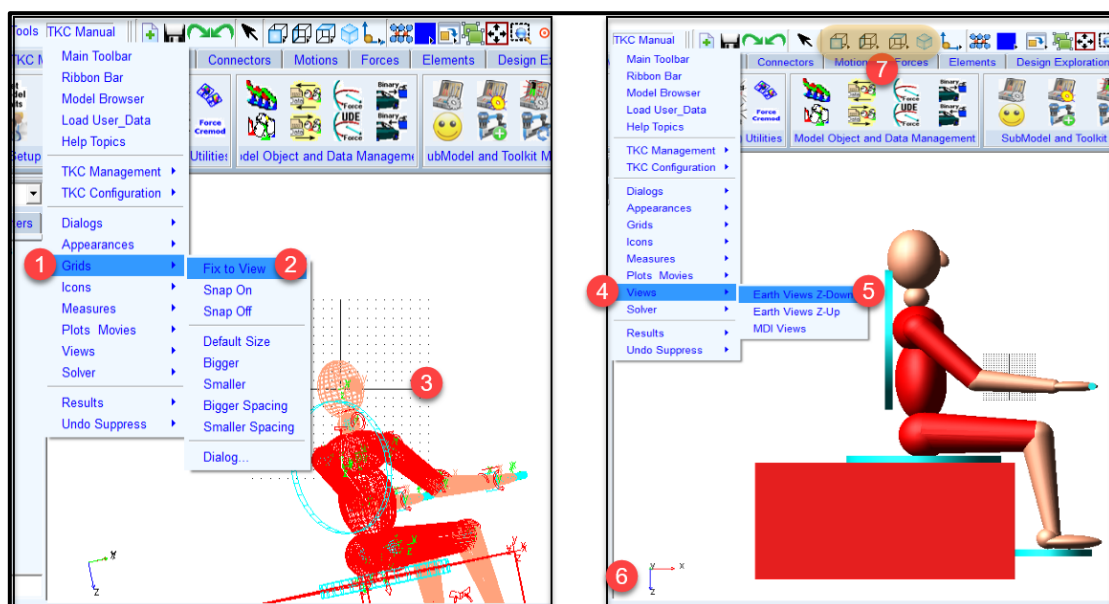
- *All*: Denotes all Toolkits known to TKC. Selecting *All* rehashes possible sources for new *User_Site* toolkits to connect;
- *Loaded*: Toolkits currently loaded in TKC;
- *Shared*: Toolkits in the TKC Shared library;
- *Site*: Toolkits in the TKC Site library;
- *User_Site*: Toolkits defined as User Toolkits, either in the *User/_Util* directory or defined in *_TKC_User_Data.cfg*

The right-side buttons on the control dialog (6) operate on the Toolkit listed in the left *Toolkit* list.

4: Main TKC Menu Macro Functions

The main menu macro functions are typically implemented as shortcuts to simplify functions and manipulations often used in daily modeling practice. Example shortcuts are for:

- *Appearances*: Buttons for showing/hiding selected types of geometry or for increasing or decreasing the graphics quality of the model geometry.
- *Grids*: Buttons for modifying or scaling the ADAMS grid used as a model sketch plane. In *Fix_to_View* mode, a personal preference, (1) the grid is fixed to the view (2), eliminating one drawing possibility. Grids buttons help to speed-up the Grid Settings dialog actions. See image below-left.



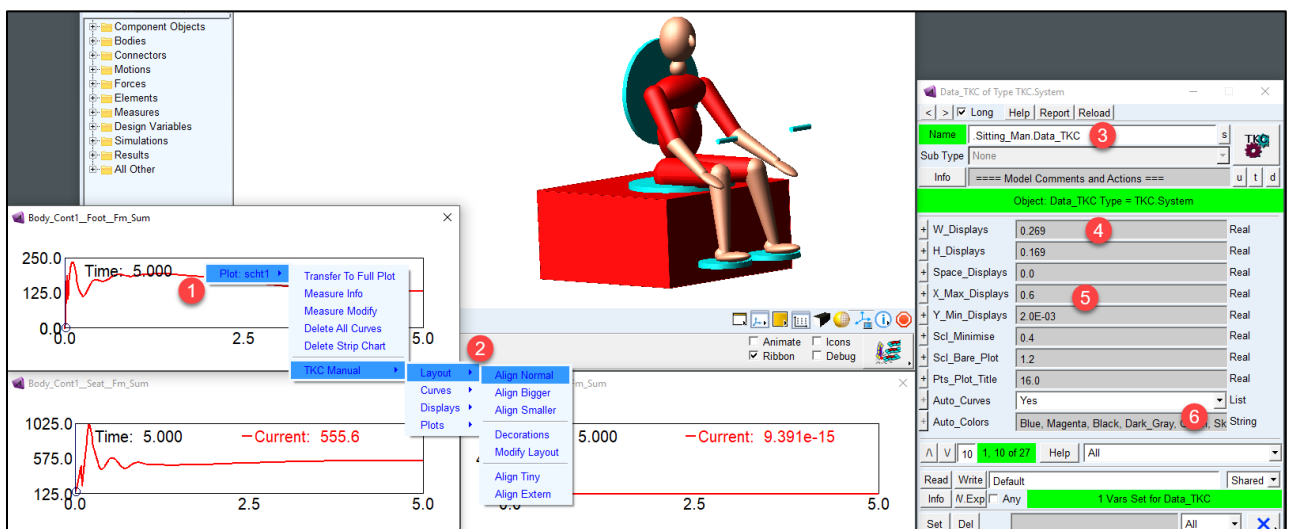
- **Icons:** Buttons intended to speed-up actions in the Icon Settings dialog. In typical use, one applies a single optimal grid size and toggles the visibility of type depending icons.
- **Measures:** Shortcuts and dialogs to show, scale, align and modify the appearance of measure displays on the simulation screen.
- **Plots_Movies:** Options and dialog to show/hide the ADAMS Post Processing environment and show a dedicated dialog for a list of strip book resembling movie displays.
- **Views:** See image above-right. The Views group (4) speeds-up mapping between the standard view orientations of the ADAMS/View Gui (7) and an overall intuitive view of the global frame (5).
- **Solver:** Change embedded and external ADAMS/Solver settings.

5: Object-Based Popup Menus

Some dedicated Popup Menus are defined in the TKC environment. Note again that each TKC Popup is labeled with the **TKC_Project_Name** which makes it easy to find all added TKC functionality. For model objects, Popup Menus appear which allow manipulation of a hierarchy of model objects, which can either be the Adams/View object primitive (i.e., an ADAMS part) or the TKC component (i.e., the total human dummy object) or the TKC object data (i.e., the data of the human dummy object). In case of hidden objects, its TKC Popup functionality is still available by a right-mouse click on fields of the object TKC component edit dialog.

6: Measure Display Popup Menu

The TKC Measure Popup Menu is displayed by (1) a right-mouse click inside any measure display is shown. It contains a range of menus and **options** for modification of (2) measure display alignment and scaling on the simulation screen, for changing measure display decorations, for saving and deleting plot curves, for storing and re-storing measure displays and creating plots from the screen displays.



The TKC system data array **[Model].Data_TKC** (3) contains several design variables to:

- (4) apply the desired measure display scaling and spacing,
- (5) apply the portion of the entire screen used for measure displays and,
- (6) define measure curve legends Colors and Labels.

Modification and storage of the TKC system data variables allow for the project depending user settings of measure displays applied.

7: Generic Toolkits Dialogs

Toolkit functionality for creating components objects and data or calling utilities is stored in a dedicated Toolkit dialog with a generic interface. All Toolkit dialogs define four sets of buttons or button stacks. Empty button stacks, in case of zero components or utilities, are reflected by a grey zone in the toolkit dialog.



From left-to-right or top-to-bottom four buttons (stacks) are defined:

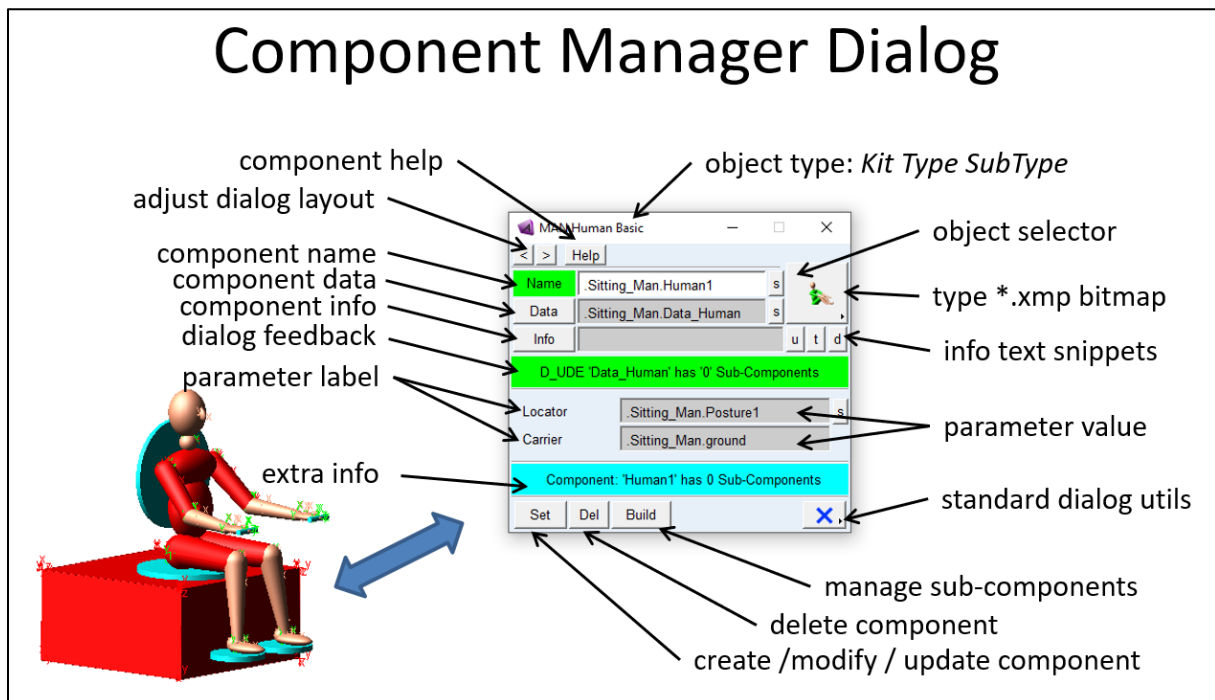
- (1) The toolkit label button toggles the dialog to be aligned either vertically or horizontally.
- (2) The component manager button stack contains a button for each model component defined in the toolkit. Each component dialog is uniquely defined by the parameters as defined in the toolkit configuration file for the component at hand. Multiple types of each component can be available as component sub-types.
- (3) The component data manager button stack contains a button for each component with data variables defined (implying that also components exist with no data variables). Note that data for a new component can also be created from the data button in the respective component manager dialog. Sub-Types of components will typically have a unique list of design variables to define component properties.
- (4) The utility button stack contains a button for each registered utility in the toolkit. Each utility in this list opens the generic utility dialog with appropriate bitmap and number of utility parameters. Note that utilities can also be *un-registered* which will prevent them from popping up in this list. Use of an un-registered utility is still possible as an *open call* from the generic utility dialog.

8: Component Object Manager Dialog

Component objects in TKC are created and modified using generic dialogs. Each dialog is automatically created from the component configuration data in the toolkit configuration file. All information is used in the layout and callback actions of the dedicated component management dialog. The image below illustrates the dialog for managing a component of type *MAN.Human.Basic*.

The behavior and functionality of some objects in the dialog will be shortly described:

- *Field labels* and dialog feedback labels will typically have a yellow color for objects that are yet to be defined (component name or component data). Once created, these labels will turn green.
- If applicable, a dedicated *object selector* button will appear at the right of an input field. These selectors are TKC sensitive and allow users to select different types of objects from existing model entities.
- Labels and field input types of component parameters are automatically created from the definition. This includes the number of input parameters of a component (i.e., markers, parts, other components) and the reaction of the component (*.mac_cre) macro after changing the value of a parameter (i.e., does the component allow for swapping a marker or does that generate a message to re-create the component).



- The appearance of the Lower left button indicates the status of the component instance. Before creation it reads *Create*, after creation, it reads *Set* or *Modifies* depending on the requirement to activate a parameter callback update.
- The SubType of the component (i.e., *Basic* for *Basic Human* object) is selected with the bitmap button stack selector. This can only be done if the component is not created yet. If a different SubType is selected, the SubType of the Data element will also be updated.
- For convenience, some utilities can be accessed directly from the dialog.

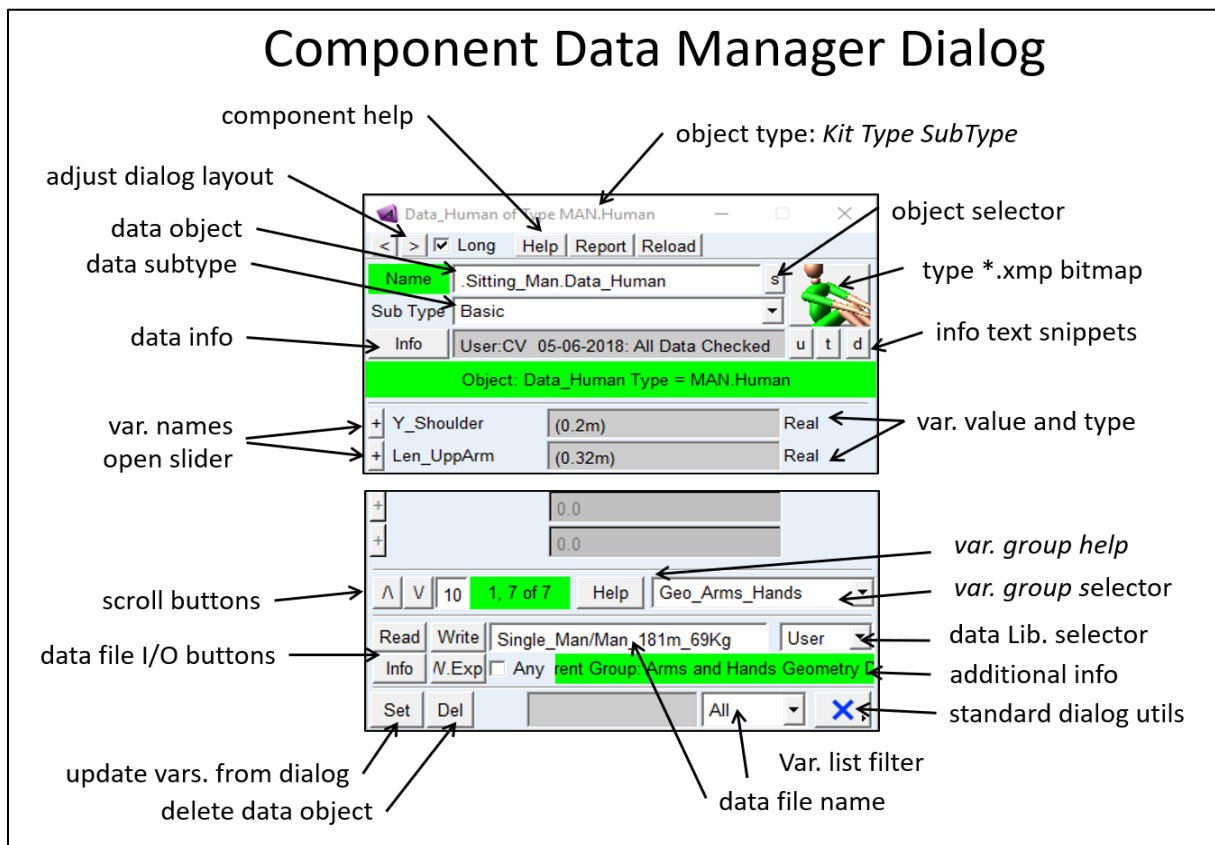
Some exercise in defining and updating a component helps to get more accustomed to the details of the component manager dialog.

9: Component Data Manager Dialog

Design variables for parametrization of TKC components are stored as children of a TKC component data object. This object, which is another design variable, is marked so that the default modify method is to display the dedicated object data manager dialog. The image below shows the top and bottom of this dialog for data of a *MAN.Human.Basic* type object.

The behavior and functionality of some objects in the dialog will be described:

- Special string type variables can be defined in the data object to define groups of variables (syntax: name is `_grp_XXX`, i.e., `_grp_Geo_Arms_Hands`). This approach enhances the readability of data objects with many variables as they can systematically be stored and viewed in groups.
- The main dialog Help at the dialog top lists the code of the data definition macro. The variable group help button will display a picture, text file or movie clip depending on the availability of these media files in the help library of the toolkit at hand. Typically, these help media files can be applied to give a detailed explanation of the component parametrization method.



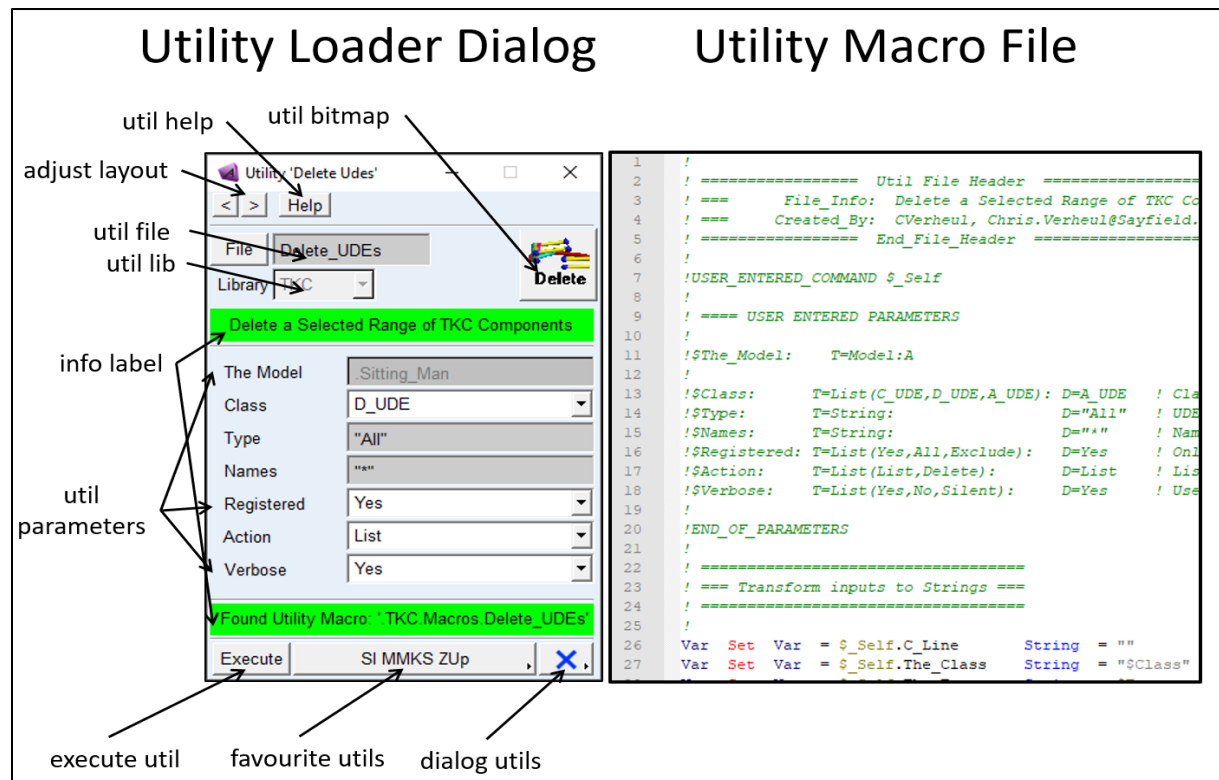
- If applicable, generic slider dialogs for scalar or vector variables be displayed from the left side **+** buttons in the dialog. These slider dialogs provide a useful interactive visual check of component parametrization.
- TKC supports data object file storage by a variety of methods. Data can be stored in a range of data libraries as well as in sub-directories of these libraries. Component data files are uniquely stored as the component type and subtype define the data file extension. As an example, data files of *MAN.Human.Basic* data objects have a file name filter of **.Human_Basic*.
- The **Any** toggle button in the data I/O group resets the data file filter to any data file. As names of data objects in these files are checked, this allows import of other component type data files.

10: Utility Loader Dialog

Utilities in TKC are defined in a library or toolkit depending on its purpose and connection to certain operations in defining and using models. They typically serve to perform modeling, simulation or post-processing operations without a direct link to model components or data objects. After selecting the proper Utility file from the Utility loader dialog, TKC will read the macro header to find and check utility parameters. Each macro parameter will create an input field in the utility dialog, so there is a direct link between the appearance of the loader and the interface of the utility macro.

The behavior and functionality of some objects in the utility loader dialog will be described:

- The image below shows the loader dialog on the left and the utility macro file on the right side. In this example, the utility loader is opened in a *closed call* as the library selection list is greyed out. When the loader is opened in a so-called *open call*, all toolkits and libraries can be browsed for requested utility macros.
- The utility file can either be selected using the file name field or, in case a utility bitmap is provided, using the utility bitmap browser. After loading a utility file in the dialog, both dialogs objects will reflect the same utility.



- The number and names of parameter fields in the utility dialog is identical to the number of parameters in the utility file.
- For utilities, no additional definition of types and number of parameters is required in the toolkit cfg file. All parameter information is extracted from the utility file header lines up to the text *End_of_Parameters*, including comment strings for quick help and extended parameter properties. The user is advised to check through the available utility macros in the SHARED and TKC libraries to get an impression of possible utility parameter features.