Offline Configuration

4.02.05 Maintaining Config Structures

Each transaction - whether Master Record or Addendum - will have a configuration screen, called a Config screen. Each config screen will correspond to a data table. The data tables contain elements which, as defined in the Properties section above, will be the Name, Pattern, Database Structure, Screen Structure and TCode Structure. These are defined as follows:

- **Name:** Each config screen will have a name. To assign a name, use the following code. Note that the [0] array number indicates that this is the Master Record's config screen - it is always the initial entry in an array.:

  ```
  g_WorkOrderAppDesc.Config[0] = "Configuration";
  ```

- **Pattern:** The Pattern differentiates between the config files and also identifies what kind of screen is created for each Addendum. Like the Master Record and Addendums, the config screens can have two types of patterns - Table or Normal. In our example, we will use the Normal style. The code is as follows:

  ```
  g_WorkOrderAppDesc.Config[0].Pattern = CONFIG_PATTERN_NORMAL_STY
  LE;
  ```

- **Database Structure:** As discussed earlier, each config transaction contains a data table. Config screens are created from the definitions contained in these data tables. For our example, we will create a Config screen for the Master Record that contains ten fields. The code is shown below:

  ```
  g_WorkOrderAppDesc.Conf
  ig[0].Database = [//Default value for this field
  {label:"Work order",displaywidth:10,fieldlength:10,fieldname:"g_
  cf_wo",defaultvalue:""},
  {label:"Plant",displaywidth:10,fieldlength:10,fieldname:"g_cf_pl
  ant",defaultvalue:"1000"},
  {label:"Work Center",displaywidth:10,fieldlength:10,fieldname:"g
  _cf_ctr",defaultvalue:"MECHANIK"},
  {label:"Functional Location",displaywidth:10,fieldlength:10,fiel
  dname:"g_cf_floc",defaultvalue:""},
  {label:"Equipment",displaywidth:10,fieldlength:10,fieldname:"g_c
  f_equno",defaultvalue:""},
  {label:"Order Type",displaywidth:10,fieldlength:10,fieldname:"g_
  cf_otype",defaultvalue:"PM01"},
  {label:"To",displaywidth:10,fieldlength:10,fieldname:"g_cf_otype
  2",defaultvalue:"",usesameline:1},
  {label:"Download Rows",displaywidth:10,fieldlength:10,fieldname:
  "g_cf_dlrows",defaultvalue:"5"},
  {label:"From Date",displaywidth:10,fieldlength:10,fieldname:"g_c
  f_frdate", defaultvalue:"01.04.2003"},
  {label:"To Date",displaywidth:10,fieldlength:10,fieldname:"g_cf_
  todate",defaultvalue:""} ];
  ```


(c) 2026 Liquid UI | Synactive | GuiXT <dev@guixt.com> | 2026-11-02 20:01

URL: https://www.liquid-ui.com/knowledge_base/content/130/1059/en/40205-maintaining-config-structures.html

Offline Configuration

- **Screen Structure:** It may be necessary to create pushbuttons to provide navigation as well as either basic or custom functionality for the transaction. We will create a single pushbutton, for which the code is as follows:

  ```
  g_WorkOrderAppDesc.Config[0].Screen = "@11@Configure";
  ```

- **Master Structure:** After creating the screen structure, we need to define the transaction for which we are creating the config screen. In this example, the transaction is 'wkdr'. The code is as follows:

  ```
  g_WorkOrderAppDesc.Config[0].Master = "wkdr";
  ```

- **TCode Structure:** The final element of the Config database is the transaction code for the transaction in question. Any data saved for the configuration screen for a particular transaction will be stored in this data table. In our example, we will specify that the transaction is 'conf'. The code is as follows:

  ```
  g_WorkOrderAppDesc.Config[0].TCode = "conf";
  ```

Unique solution ID: #2062
Author: sarvani.kusuri@guixt.com
Last update: 2021-06-03 17:12

(c) 2026 Liquid UI | Synactive | GuiXT <dev@guixt.com> | 2026-11-02 20:01
URL: https://www.liquid-ui.com/knowledge_base/content/130/1059/en/40205-maintaining-config-structures.html