



# TypeLists



June 1, 2014

© Guidewire Software, Inc. 2001-2014. All rights reserved.  
Do not distribute without permission.

Guidewire training materials contain Guidewire proprietary information that is subject to confidentiality and non-disclosure agreements. You agree to use the information in this manual solely for the purpose of training to implement Guidewire software solutions. You also agree not to disclose the information in this manual to third parties or copy this manual without prior written consent from Guidewire. Guidewire training may be given only by Guidewire employees or certified Guidewire partners under the appropriate agreement with Guidewire.

# Lesson objectives

- By the end of this lesson, you should be able to:
  - Describe the functionality of a typelist
  - Create a typelist and typelist extension
  - Create a typekey field
  - Create a typekey field that references a typelist filter

This lesson uses the notes section for additional explanation and information.  
To view the notes in PowerPoint, select View → Normal or View → Notes Page.  
When printing notes, select Note Pages and Print hidden slides.

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

2

G U I D E W I R E



## Lesson outline

- Typelist basics
- Create a typelist
- Create a typelist extension
- Defining typekey fields

# What is a typelist?

- A typelist is a list of defined values
- Defined values often used to constrain user input
- When rendered in UI, typically appears as dropdown list

The image shows two side-by-side windows. On the left is a user interface window titled 'Account Type' with a dropdown menu. The dropdown shows three options: 'Checking', 'Savings', and 'Other'. The 'Savings' option is highlighted with a red box and a red arrow points from it to the corresponding row in the typelist table on the right. On the right is a tool window titled 'BankAccountType.tti' containing a table with four columns: Element, Code, Name, and Priority. The table has one row for 'typelist' and three rows for 'typecode'. The 'typelist' row has 'BankAccountType' in the 'Code' column and 'The type of bank ac...' in the 'Name' column. The three 'typecode' rows have 'checking', 'savings', and 'other' in the 'Code' column, and 'Checking', 'Savings', and 'Other' in the 'Name' column respectively. The 'Priority' column shows values 1, 2, and 3 for each 'typecode' row.

| Element  | Code            | Name                   | Priority |
|----------|-----------------|------------------------|----------|
| typelist | BankAccountType | The type of bank ac... |          |
| typecode | checking        | Checking               | 1        |
| typecode | savings         | Savings                | 2        |
| typecode | other           | Other                  | 3        |

BankAccountType typelist defines values in Account Type cell

# Locations of typelist files

... \Metadata\Typelist



Typelist

Internal  
Typelist  
Extension

... \Extensions\Typelist



Typelist

Typelist  
Extension

- Most platform and base application typelists
- All files are read-only
- Some files marked as final
- Some base application typelists and typelist extensions
- Customer custom typelists and typelist extensions
- All files are editable

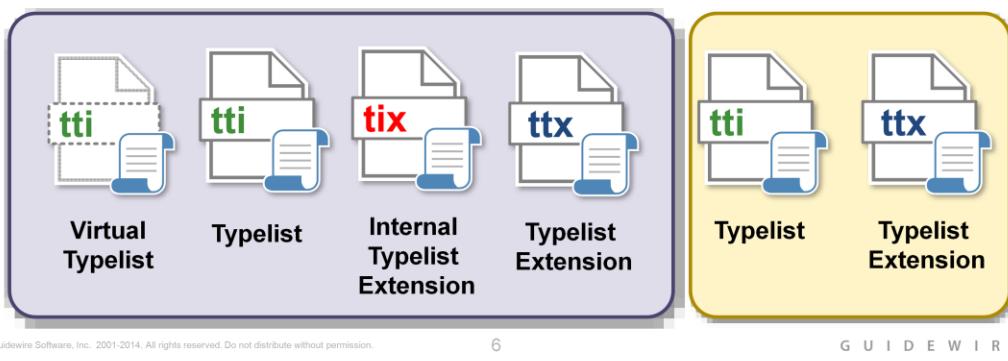
© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

5

G U I D E W I R E

# Kinds of typelists

- Internal
  - Internal base application logic
  - Cannot modify or extend
  - Read-only
  - Some virtual
- Extendable
  - Base application
  - Can modify typelist or typelist extension
- Custom
  - Customer creates as part of custom configuration



In some cases, an internal typelist does not have a physical file that is exposed in Guidewire Studio. One type of virtual typelist is a typelist of subtypes for a given supertype. In Guidewire Studio, using Class Name Search (CTRL+N), you can find and view a virtual typelists in the Typelist Editor. However, there is no file path and no XML file view for virtual typelists. One example is the ABContact typelist which is a typelist of all ABContact subtypes.

Typelists with the final="true" attribute defined in the <typelist /> element cannot be extended. An internal typelist extension (TIX) file can only be found in the ...\\modules\\configuration\\config\\metadata\\typelist\\ folder.

You can always determine to base configuration for a Guidewire application by looking at the <installDirectory>\\modules\\base.zip file.

# Internally managed (virtual) typelists

- Some typelists are not exposed
- No physical file exposed
  - Typelist Editor file tab shows yellow for virtual status
  - Typelist Editor XML tab shows empty file
  - Studio navigation bar is blank
- Example: Subtype typelist
  - Shows all subtypes a specific subtyped parent entity
  - New subtypes entities automatically added

Virtual  
Typelist

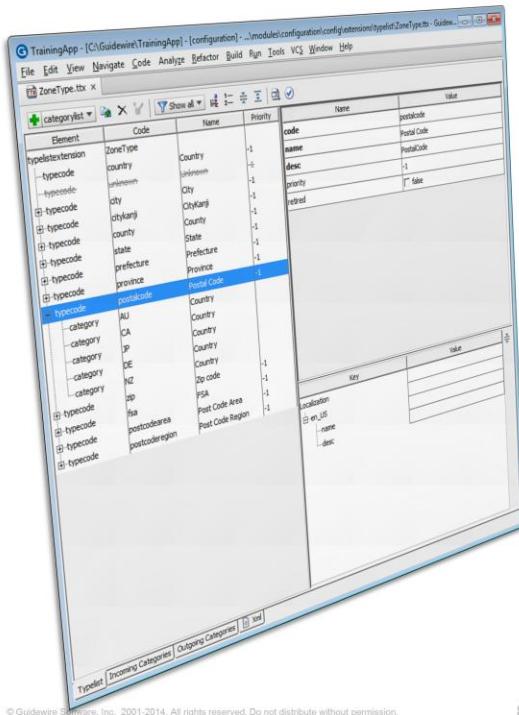
| Element  | Code                 | Name                    | Priority |
|----------|----------------------|-------------------------|----------|
| typelist | ABContact            | Subtype typelist for... |          |
| typecode | ABContact            | Contact                 | -1       |
| typecode | ABPerson             | Person                  | -1       |
| typecode | ABCompany            | Company                 | -1       |
| typecode | ABPlace              | Place                   | -1       |
| typecode | ABLLegalVenue        | Legal Venue             | -1       |
| typecode | ABPolicyCompany      | Policy Company          | -1       |
| typecode | ABUserContact        | User Contact            | -1       |
| typecode | ABPersonVendor       | Vendor (Person)         | -1       |
| typecode | ABCCompanyVendor     | Vendor (Company)        | -1       |
| typecode | ABAdjudicator        | Adjudicator             | -1       |
| typecode | ABPolicyPerson       | Policy Person           | -1       |
| typecode | ABAutoTowingAgcy     | Auto Towing Agency      | -1       |
| typecode | ABDoctor             | Doctor                  | -1       |
| typecode | ABMedicalCareOrg     | Medical Care Organi...  | -1       |
| typecode | ABAAutoScrapYard_Ext | Auto Scrap Yard         | -1       |
| typecode | ABAAutoRepairShop    | Auto Repair Shop        | -1       |
| typecode | ABLawFirm            | Law Firm                | -1       |
| typecode | ABAAttorney          | Attorney                | -1       |

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

7

G U I D E W I R E

# Typelist editor



- View, define and edit a typelist, typelist extension, or internal typelist extension
  - **/extensions/typelist/**
    - TTI and TTX files
  - **/metadata/typelist/**
    - TTI and TIX files
- Consists of
  - Editor toolbar
  - Element tree pane
  - Attribute pane
  - Localization pane
- Metadata Editor is base editor

8

GUIDEWIRE

With the Typelist editor, you define a typelist and any associated typecodes, typefilters, and categorylists. With the Typelist editor, you can view all (but not edit all) typelists: extendable, internal, internally exposed (typelist of subtypes, for example), and custom typelists.

You can define and edit an typelist and typelist extension located in the ...\\configuration\\config\\extensions\\typelist folder. The Typelist editor also allows you to view an typelist or internal typelist extension located in the ...\\configuration\\config\\metadata\\typelist folder. Files in the metadata folder are read-only.

# Typelist editor: Toolbar reference

|          | Icon     | Description                                   |  |
|----------|----------|---|--|
| Actions  | Edit     | Add an element; Dropdown list is schema aware |  |
|          |          |   | Duplicate selected element                       |
|          |          |   | Delete selected element                          |
|          |          |   | Override selected element                        |
|          | View     |   | Filter elements by file                          |
|          |          |   | Persist sort order                               |
|          |          |   | Reassign priorities                              |
|          |          |   | Collapse nested elements; Expand nested elements |
|          | Navigate |   | Navigate to supertype and/or subtype             |
|          |          | <a href="#">ZoneType.ttx</a>                  | Click link to navigate to extension              |
| Validate |          | Validate typelist                             |  |

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

9

G U I D E W I R E

When you define and edit a typelist or typelist extension located in the /configuration/config/extensions/typelist folder, the Typelist editor toolbar is fully enabled. Typelists and internal typelist extensions in the /configuration/config/metadata/entity folder are read-only and all Edit actions are disabled.

The Typelist editor toolbar is also context sensitive based on the kind of typelist (internal, internally exposed, extendable, or custom). For internal typelists, there are no edit actions in the toolbar.

The Typelist editor toolbar is also context sensitive based on the bottom tab selection. For both the Incoming Categories and Outgoing Categories bottom tabs, the toolbar consists only of the Add Category/Typecode, Remove, Filter, Collapse, Expand, and Validate commands.

# TypeList editor: Element tree pane

| Element           | Code       | Name        | Priority |
|-------------------|------------|-------------|----------|
| typelistextension | ZoneType   |             |          |
| typecode          | country    | Country     | -1       |
| typecode          | unknown    | Unknown     | -1       |
| typecode          | postalcode | Postal Code |          |
| category          | AU         | Country     |          |
| category          | CA         | Country     |          |
| category          | JP         | Country     |          |

- Displays hierarchy of XML elements
  - Elements merged from underlying base typelists are read-only
  - Sortable columns
- Context menu is schema aware
  - Add new elements as siblings and children
  - Cut, Copy, and Paste elements

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

10

G U I D E W I R E

View actions in the TypeList editor toolbar influence the display of the Element tree pane. Bottom tab selection also changes the columns of the Element tree pane for the TypeList editor. For instance, Incoming Categories and Outgoing Categories changes the columns to Category/Typecode and TypeList. By default, the Element tree pane displays the hierarchy of nested elements, including any inherited elements from base typelists. You can use the toolbar to change how the editor display the various elements: Show all, This file only, Typecodes only, and Typefilters only. The toolbar and the context menu for the element tree pane is schema aware.

## TypeList editor: Attribute pane

- For selected element, define attributes
  - Name is the attribute
  - Value is the attribute value

| Name     | Value                          |
|----------|--------------------------------|
| code     | postalcode                     |
| name     | Postal Code                    |
| desc     | PostalCode                     |
| priority | -1                             |
| retired  | <input type="checkbox"/> false |

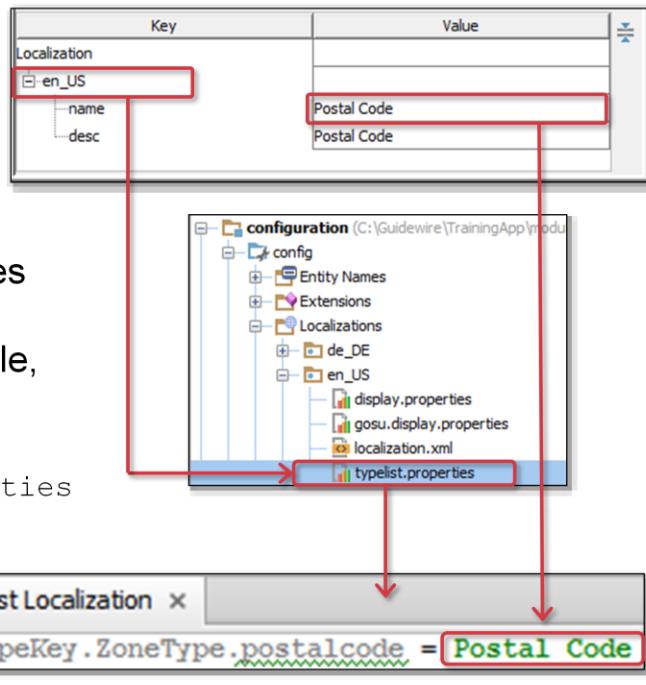
<typecode  
code="postalcode"  
name="Postal Code"  
desc="PostalCode">

- Attribute styling
  - Bold for required and black for editable
  - Grayed-out for non-editable
  - Overridden, Inherited, Internal and Default

# Localization pane

- Localize a specific typecode
  - Name
  - Description

- New or edited values modified in the typelist.properties file, example
  - /locale/en\_US/typelist.properties



© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

12

G U I D E W I R E

Use the Localization pane to select a specific locale for a given typecode. Then, create or edit a translation string for both the typecode name and description. The name and description that you specify is created in the typelists.properties file for the locale. Guidewire stores typelist typekey localizations in typelist.properties files, one file for each locale. Guidewire Studio stores typelist typekey localizations using the following format for name and description respectively:

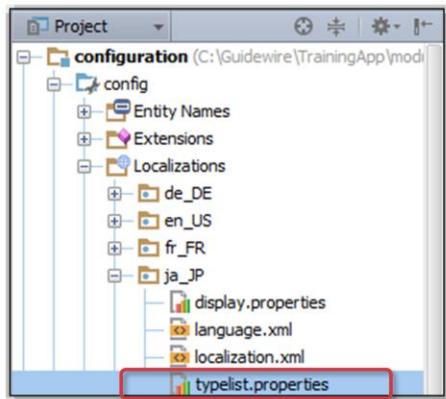
TypeKey.typekey = localized\_name

TypeKeyDescription.typekey = localized\_description

You can edit typecode name and descriptions in the Typelist editor Localization pane or in the Typelist Localization editor.

# TypeList Localization text editor

- For a given locale, edit typelist.properties
- Define TypeKey and/or TypeKeyDescription
- In file search
  - CTRL+F
- Remove and sort
  - ALT+ENTER



© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

13

G U I D E W I R E

Guidewire stores typelist typekey localizations in typelist.properties files, one file for each locale. Guidewire Studio stores typelist typekey localizations using the following format for name and description respectively:

- TypeKey.typeList.typekey = localized\_name
- TypeKeyDescription.typeList.typekey = localized\_description

You can edit typecode name and descriptions in the Typelist editor Localization pane or in the Typelist Localization editor.

There are several ways to view localized typekey properties:

- typekey.Code = String that represents the typecode
- typekey.DisplayName = Localized language version of the entity name
- typekey.UnlocalizedName = Name listed in the data model

## Typecodes for base application typelists

| Element  | Code            | Name                         | Priority |
|----------|-----------------|------------------------------|----------|
| typelist | BankAccountType | The type of bank account ... |          |
| typecode | checking        | Checking                     | 1        |
| typecode | savings         | Savings                      | 2        |
| typecode | other           | Other                        | 3        |
| typecode | retirement_Ext  | Retirement                   | 4        |

| Name     | Value                          |
|----------|--------------------------------|
| code     | retirement_Ext                 |
| name     | Retirement                     |
| desc     | Retirement                     |
| priority | 4                              |
| retired  | <input type="checkbox"/> false |

- When extending base application typelists or typelist extensions in the ...\\**Extensions\\Typelist\\** folder, create new typecodes with codes that end in \_Ext

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

14

G U I D E W I R E

The BankAccountType.tti is an application typelist found in the ...\\modules\\configuration\\config\\extensions\\typelist\\ folder.

For TTI and TTX files that are extendable (not marked as final) and editable in the \\extensions\\typelist\\ folder that are part of the base application, use the \_Ext naming convention by adding \_Ext to the name of the typecode.



## Lesson outline

- Typelist basics
- Create a typelist
- Create a typelist extension
- Defining typekey fields

## Steps to create a typelist

1. Create the typelist file
2. Define typecodes
  - a) Optionally Localize typecode descriptions
3. Optionally regenerate the dictionary
4. Deploy the typelist

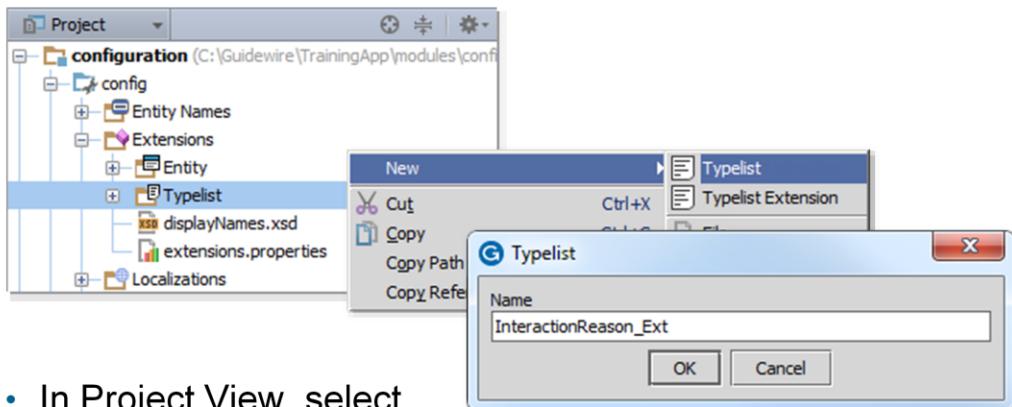
© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

16

G U I D E W I R E

To create an typelist, in Project view, navigate to the /configuration/config/Extensions/Typelist folder. Open the folder context menu. Select Typelist. In the Typelist dialog, name the typelist. Click OK.

## Step 1: Create the typelist file



- In Project View, select .../config/Extensions/Typelists
  - Context menu → New → Typelist
- Typelist dialog
  - Naming convention is for custom typelists to end in `_Ext`
  - Example: `InteractionReason_Ext`

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

17

G U I D E W I R E

The maximum length for a typelist name is 25 characters.

By default, Guidewire uses `xxtl_<typelist name>` as the name of the typelist table (where "xx" is the application's code). If you want a different table name, you can override the default value by specifying a value in the "Table name" field in Studio. If you override the default value, the table name becomes `xxtl_<table name>`.

Guidewire restricts the typelist table name to ASCII letters, digits, and the underscore character. Guidewire also places limits on the length of the name.

Guidewire imposes these restrictions because of the restrictions imposed by the various relational database management systems (RDBMS) with which Guidewire applications interact. If you choose, however, you can override the name of the typelist and thus the table name stored in the database.

A "final" typelist is a typelist to which no additional typecodes can be added. All custom typelists are non-final.

## Step 2: Define typecodes

The screenshot shows the Guidewire Typelist Editor interface. The main table lists typecodes under a typeplist named 'InteractionReason\_Ext'. A specific typecode row for 'other' is selected. To the right, a detailed view pane shows attributes for this typecode, including 'code' (value 'other'), 'name' (value 'Other'), 'desc' (value 'Other'), 'priority' (value '50'), and 'retired' (checkbox checked). A red box highlights the 'priority' row.

| Element   | Code                  | Name                  | Priority |  |
|-----------|-----------------------|-----------------------|----------|--|
| typeplist | InteractionReason_Ext | InteractionReason_Ext |          |  |
| typecode  | request               | Request               | 10       |  |
| typecode  | complaint             | Complaint             | 20       |  |
| typecode  | courtesyContact       | Courtesy Contact      | 30       |  |
| typecode  | other                 | Other                 | 50       |  |

| Name     | Value                                     |
|----------|---|
| code     | other                                     |
| name     | Other                                     |
| desc     | Other                                     |
| priority | 50  |
| retired  | <input checked="" type="checkbox"/> false |

- Add new typecodes with the toolbar or context menu
- Define typecode attributes in the Name Value pane
  - Code is an internal reference, must be unique within the typelist, <=50 characters and contain alphanumeric values
  - Name is the default for when displayed in the user interface
  - Desc is what the data dictionary displays
  - Priority is the sort order
  - Retired prevents new objects from using the typecode

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

18

G U I D E W I R E

Use the Typelist Editor to add typecode, categories, and filters.

If a list view is sorted based on a typekey field, then the rows are sorted using standard typecode sorting. Standard typecode sorting is first by priority and then by name, and items with a priority of -1 appearing at the beginning of the list.

In a production environment, you should never delete a typecode or modify the typecode's code value. Either of these behaviors can cause problems because there could be existing objects that reference the altered or deleted typecode. In situations like this, you should mark the typecode's retired value as true. Retired=true preserves the typecode for objects that already make use of the typecode, and prevents the use of the typecode by new objects or by existing objects that don't already use the typecode. For example, assume you have a Color typelist that is used by the Vehicle entity and you retire the typecode brown. Any existing vehicle with the color brown will be unaffected. But new vehicles cannot be set to brown and existing vehicles that are not brown cannot be changed to brown.

## Validate the schema



Click Validate in the toolbar

- Red highlight indicates schema violation warning
- Schema violations explained in pane below editor

The screenshot shows the Guidewire schema editor interface. The main window displays a table for a 'typecode' list under the 'InteractionReason\_Ext.tti' file. The table has columns: Element, Code, Name, Priority, and a secondary row for properties. The first row is highlighted in red, indicating a schema violation. The properties row shows 'code' with value 'other', 'name' with value 'Other', 'desc' with value 'Other', 'priority' with value '50', and 'retired' with a checked checkbox labeled 'false'. Below the table are tabs for 'Typelist', 'Incoming Categories', 'Outgoing Categories', and 'XML'. A status bar at the bottom shows the message: 'TypelistCodeValidator - Typelist "InteractionReason\_Ext" contains multiple definitions for code "other".' The toolbar at the top includes icons for validate, save, and other operations.

| Element  | Code               | Name               | Priority |  |
|----------|--------------------|--------------------|----------|--|
| typecode | InteractionReas... | InteractionReas... |          |  |
| typecode | request            | Request            | 10       |  |
| typecode | complaint          | Complaint          | 20       |  |
| typecode | courtesyContact    | Courtesy Contact   | 30       |  |
| typecode | other              | Other              | 50       |  |
| typecode | other              | Other              | 50       |  |

Typecode Properties:

| Name     | Value                                     |
|----------|---|
| code     | other                                     |
| name     | Other                                     |
| desc     | Other                                     |
| priority | 50  |
| retired  | <input checked="" type="checkbox"/> false |

Localization:

TypelistCodeValidator - Typelist "InteractionReason\_Ext" contains multiple definitions for code "other".

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

19

G U I D E W I R E

## Step 3: Optionally regenerate dictionary

- **gwXX regen-dictionary**
- Process builds entire entity model including base and custom typelists
- Identifies errors in the data model beyond Typelist Editor schema validation

```
C:\Guidewire\TrainingApp\bin>gwta regen-dictionary  
regen-entity-model-xml:  
=====  
= Running main class:  
  com.guidewire.tools.dictionary.data.EntityModelXmlTool  
  [java] --- Guidewire Entity Model In Xml ---  
...  
ERROR Errors found in DoctorSpecialtyType  
ERROR TypelistCategoriesValidator - Typecode  
  DoctorSpecialtyType.Critical Care Medicine refers to a  
  non-existent category
```

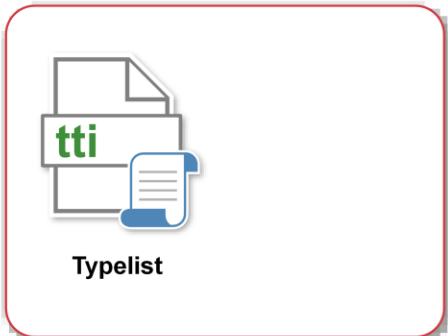
Regenerating the data dictionary is not required, but doing so can identify flawed XML in the data model that go beyond schema validation. For example, a deleted typecode that is referenced by other typelists or entities will throw an error.

## Step 4: Deploy the typelist

Restart Server

- Typelist

- bin command window
  - `gwXX dev-stop`
  - `gwXX dev-start`
- Or, Guidewire Studio
  - Run → Stop
  - Run 'Server' or Debug 'Server'
- During start-up
  - If `autoupgrade=true` in `database-config.xml`, then Guidewire attempts to upgrade the database according to the changes in the data model



© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

21

G U I D E W I R E

In Studio, the newly created typelist is available to reference immediately. However, the typelist is not available to the application server until the server is restarted.



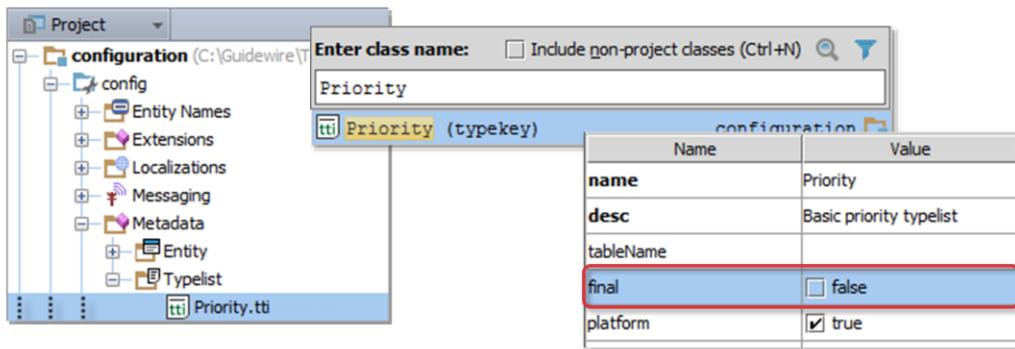
## Lesson outline

- Typelist basics
- Create a typelist
- Create a typelist extension
- Defining typekey fields

## Steps to create a typelist extension

1. Navigate to the typelist
2. Create a typelist extension file
3. Define typecodes
  - a) Optionally Localize typecode descriptions
4. Optionally regenerate the dictionary
5. Deploy the typelist extension

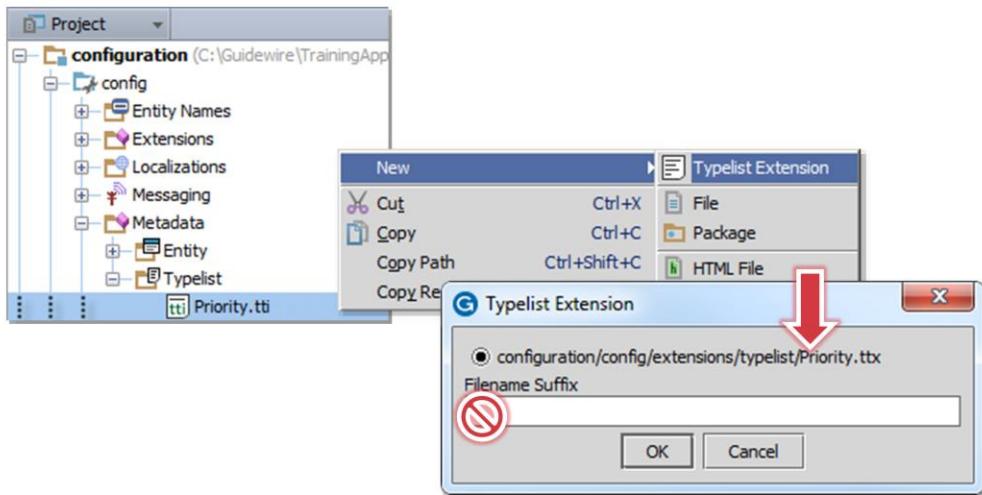
# Step 1: Navigate to a typelist



- Navigate to a typelist in ...\\Metadata\\Typelist\\
  - Project View or using CTRL+N
  - Studio will open an existing extension first!
    - If an existing extension already exists, STOP
    - Do NOT create a new extension; edit existing extension instead!
- Typelist (TTI) file must **NOT** be final=true

## Step 2: Create a typelist extension file

- Project View → Context menu → New → Typelist Extension
- Do **NOT** enter filename suffix; Click OK

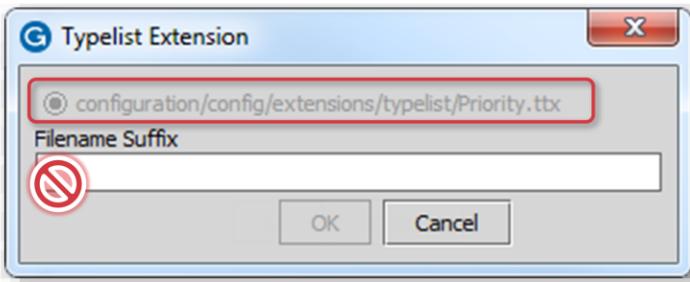


© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

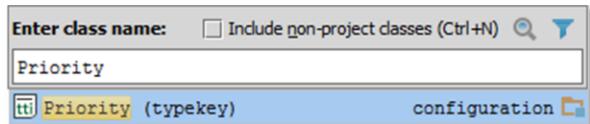
25

G U I D E W I R E

## Step 2: Create a typelist extension file (2)



- Unable to click OK? Click Cancel
  - Grayed out path shows that a typelist extension (TTX) file already exists!
- Navigate to the typelist (CTRL+N)
  - Studio automatically opens the extension file first!
  - Edit the extension in the Typelist Editor



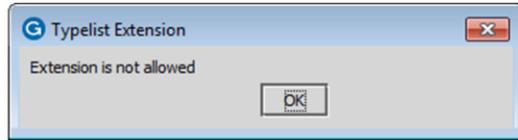
© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

26

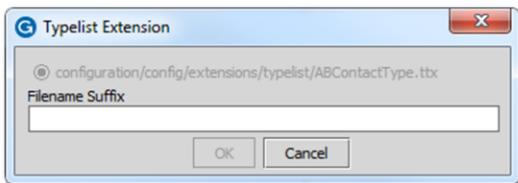
G U I D E W I R E

## Step 2: Create a typelist extension file (3)

- For a typelist (TTI) in ...\\Extensions\\Typelist\\
  - NOT possible to create typelist extension (TTX)
  - Edit the TTI file directly in the Typelist Editor
  -



- For a typelist extension (TTX) in ...\\Extensions\\Typelist\\
  - Do NOT create an extension for an extension
  - Click Cancel
  - Edit the TTX file directly in the Typelist Editor



© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

27

G U I D E W I R E

It is not possible to create a typelist extension (TTX) for a typelist (TTI) that already exists in the /Extensions/typelist/ folder. Simply edit the existing TTI file.

## Step 3: Define typecodes

The screenshot shows the Guidewire Typecode editor interface. The main window displays a table of typecodes under the heading 'typecode'. The columns are 'Element', 'Code', 'Name', and 'Priority'. The rows include 'typelistextension' (Priority), 'typecode' (urgent, high, normal, low, lowest), and 'typecode' (Normal). The 'Normal' row is selected. To the right of the main table is a secondary table for overrides, with columns 'Name' and 'Value'. It contains entries for 'code' (normal), 'name' (Normal), 'desc' (Normal), 'priority' (3), and 'retired' (false). At the bottom of the editor are tabs for 'Typelist', 'Incoming Categories', 'Outgoing Categories', and 'XML', with 'Typelist' being the active tab.

- Typelist (TTI) typecodes are in grey
  - Can change with from Show all to This file only
- Add new typecodes with toolbar or context menu to
- To change or remove typecodes, override element
  - Font color for overridden elements turn from grey to black



© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

28

G U I D E W I R E

In the slide example, the low typecode has been overridden.

## Validate the schema



Click Validate in the toolbar

- Red highlight indicates schema violation warning
- Schema violations explained in pane below editor

The screenshot shows the Guidewire XML Editor interface. The main window displays a table of typecodes with columns: Element, Code, Name, and Priority. A tooltip at the bottom of the table indicates a validation error: "TypelistCodeValidator - Typelist 'Priority' contains multiple definitions for code 'low'." The status bar at the bottom left shows copyright information: "© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission." The status bar at the bottom right shows the page number "29" and the word "GUIDEWIRE".

| Element           | Code     | Name                    | Priority |  | Name     | Value                          |
|-------------------|----------|-------------------------|----------|--|----------|--------------------------------|
| typelistextension | Priority | Basic priority typelist |          |  | code     | normal                         |
| typecode          | urgent   | Urgent                  | 1        |  | name     | Normal                         |
| typecode          | high     | High                    | 2        |  | desc     | Normal                         |
| typecode          | normal   | Normal                  | 3        |  | priority | 3                              |
| typecode          | low      | Low                     | 40       |  | retired  | <input type="checkbox"/> false |
| typecode          | low      | Low                     | 50       |  |          |                                |

## Step 4: Optionally regenerate dictionary

- **gwXX regen-dictionary**
- Process builds entire entity model including base and custom typelists
- Identifies errors in the data model beyond Typelist Editor schema validation

```
C:\Guidewire\TrainingApp\bin>gwta regen-dictionary  
regen-entity-model-xml:  
=====  
= Running main class:  
  com.guidewire.tools.dictionary.data.EntityModelXmlTool  
  [java] --- Guidewire Entity Model In Xml ---  
...  
ERROR Errors found in Priority  
ERROR TypelistCategoriesValidator - Typecode "Priority" contains  
multiple definitions for code "low".
```

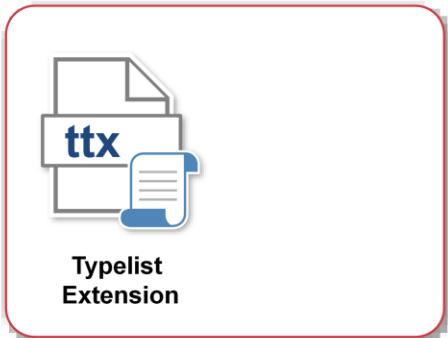
Regenerating the data dictionary is not required, but doing so can identify flawed XML in the data model that go beyond schema validation. For example, a deleted typecode that is referenced by other typelists or entities will throw an error.

## Step 5: Deploy the typelist extension

Restart Server

- Typelist extension

- bin command window
  - `gwXX dev-stop`
  - `gwXX dev-start`
- Or, Guidewire Studio
  - Run → Stop
  - Run 'Server' or Debug 'Server'
- During start-up
  - If `autoupgrade=true` in `database-config.xml`, then Guidewire attempts to upgrade the database according to the changes in the data model



© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

31

G U I D E W I R E

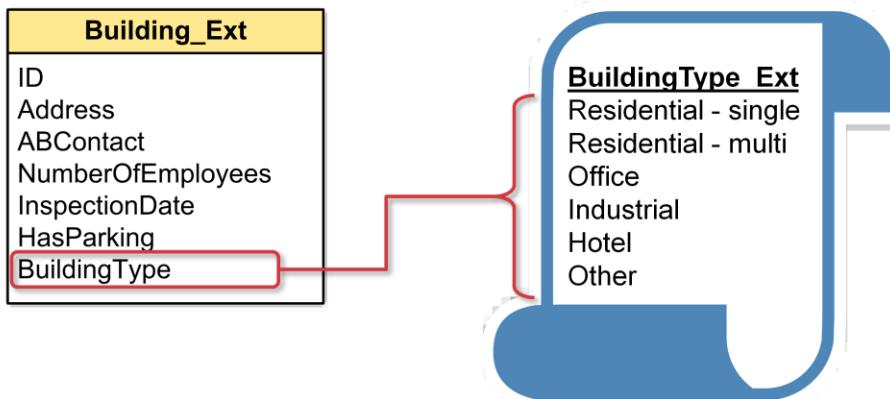
In Studio, the newly created typelist extension is available to reference immediately. However, the typelist extension is not available to the application server until the server is restarted.

## Lesson outline

- Typelist basics
- Create a typelist
- Create a typelist extension
- Defining typekey fields

## Typekey fields

- A **typekey field** is an entity defined field associated with a specific typelist
- Referenced typelist contains typecodes whose values are the only possible value for the typekey field



© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

33

G U I D E W I R E

## Entity configuration: typekey

- Add the typekey element to entity or entity extension
- Define the typelist attribute with specified typelist

| Element        | Primary Value     | Secondary Value         | Name                   | Value                                    |
|----------------|-------------------|-------------------------|------------------------|--|
| entity         | Building_Ext      | Information about a ... | <b>name</b>            | BuildingType                             |
| foreignkey     | Address           | Address                 | <b>typelist</b>        | BuildingType_Ext                         |
| foreignkey     | ABContact         | ABContact               | <b>nullok</b>          | <input checked="" type="checkbox"/> true |
| column         | NumberOfEmployees | integer                 | <b>desc</b>            | Building type                            |
| column         | InspectionDate    | datetime                | <b>columnName</b>      |  |
| column         | HasParking        | bit                     | <b>createhistogram</b> | <input type="checkbox"/> false           |
| <b>typekey</b> | BuildingType      | BuildingType_Ext        | <b>default</b>         |  |
|                |                   |                         | <b>typefilter</b>      |  |

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

34

G U I D E W I R E

## TypeList filters

- A **TypeList filter** defines a subset of typecodes in the TypeList
- Configure filters for TypeLists to filter available typecodes
- Example: YesNoOnly typefilter

The screenshot shows the Guidewire configuration interface for a TypeList filter named 'YesNoOnly'. The main window displays two tables: one for the TypeList and one for the Typefilter.

**TypeList Table:**

| Element  | Code    | Name                | Priority |
|----------|---------|---------------------|----------|
| typelist | YesNo   | Yes, No, or Unknown |          |
| typecode | yes     | Yes                 | 10       |
| typecode | no      | No                  | 20       |
| typecode | unknown | Unknown             | 30       |

**Typefilter Table:**

| Name       | Value                          |
|------------|--------------------------------|
| name       | YesNoOnly                      |
| desc       | Only display Yes and No...     |
| includeAll | <input type="checkbox"/> false |

At the bottom of the interface, there are tabs for 'TypeList', 'Incoming Categories', 'Outgoing Categories', and 'Xml'. The 'TypeList' tab is selected.

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

35

G U I D E W I R E

It is possible to configure a TypeList so that a Guidewire application can filter the TypeList values so that they do not all appear in the drop-down list (TypeList) in the application user interface.

A TypeList filter causes the TypeList to display only a subset of the typecodes for that TypeList. A filter narrows the list of typecodes to show in the TypeList view in the application.

# Create a typelist filter

The screenshot shows the Guidewire Typefilter Editor interface. On the left, a sidebar menu lists 'category', 'typecode', 'typefilter' (which is selected), and three options under 'typefilter': 'Add Categories...', 'Include Into Filter...', and 'Exclude From Filter...'. A red arrow points down from the 'typefilter' item to the main editor area. In the center, there's a table titled 'typelist' with columns 'Element', 'Code', 'Name', and 'Pri...'. Below this table is another table titled 'typefilter' with columns 'Element', 'Code', 'Name', and 'Pri...'. The 'typefilter' table has two rows: 'include yes' (selected) and 'include no'. To the right of these tables is a 'code' dropdown menu with three options: 'no', 'unknown', and 'yes'. A red box highlights the 'code' dropdown and its options.

- Add typefilter element
- Add Include Into Filter typecodes
- Specify code attribute values

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

36

G U I D E W I R E

You define a typelist filter at the level of the typelist. Use the Typelist Editor to define a filter for a particular typelist.

## Referencing a typelist filter

- For the typekey element in the entity file, specify the typefilter attribute
- Only values in the typefilter are specified (included or excluded)

The screenshot shows the Guidewire Studio interface with the Entity editor open for the ABPolicyPerson.entity file. A context menu is displayed over the typekey row in the table, with the 'YesNo.tti' tab selected. The context menu table contains the following data:

| Element    | Code      |
|------------|-----------|
| typelist   | YesNo     |
| typecode   | yes       |
| typecode   | no        |
| typecode   | unknown   |
| typefilter | YesNoOnly |
| include    | yes       |
| include    | no        |

Red boxes highlight the 'typekey' row in the main table and the 'typefilter' row in the context menu table. Red arrows point from the highlighted rows in the main table to the corresponding rows in the context menu table.

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

37

G U I D E W I R E

# Typecode category

The screenshot illustrates the Guidewire Typecode Manager interface. It consists of two main windows:

- Left Window (categorylist):** A table view showing typecodes and their properties. One row is selected for 'typecode CriticalCareMedicine' with the value 'DoctorCategoryType'.
- Right Window (DoctorCategoryType.tti):** A detailed view of a specific typecode. The 'Incoming Categories' tab is selected, showing categories like 'GeneralCare', 'ImmediateCare', and 'CriticalCareMedicine'. The 'CriticalCareMedicine' category is highlighted with a red box and has a red arrow pointing from it to the selected row in the left window.

- Category

- Associate one or more typecodes on a parent typelist with one or more typecodes on a child typelist
- Incoming Categories tab in parent typelist shows child typecodes

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

38

G U I D E W I R E

A typecode filter uses categories and category lists at the typecode level to restrict or filter a typelist.

# Dependent Drop-down lists

## Category

|                       |
|-----------------------|
| <none>                |
| <b>Immediate Care</b> |
| General Care          |
| Surgery               |
| Long-Term Care        |
| Unspecified           |

## Specialty

|                               |
|-------------------------------|
| <none>                        |
| <b>Critical Care Medicine</b> |
| Dentistry/Oral Surgery        |
| Emergency Medicine            |
| Hospitalist                   |
| Unspecified                   |

- Use categories and category lists at the typecode level to restrict or filter dependent drop-down lists

## Category

|                       |
|-----------------------|
| <none>                |
| Immediate Care        |
| General Care          |
| Surgery               |
| <b>Long-Term Care</b> |
| Unspecified           |

## Specialty

|                               |
|-------------------------------|
| <none>                        |
| <b>Critical Care Medicine</b> |
| Pain Management               |
| Physical Rehabilitation       |
| Unspecified                   |

- Example

- Doctor Category filters Doctor Specialty
- Parent filters can share children typecode

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

39

G U I D E W I R E

The entity ABDoctor specifies that the DoctorSpecialtyType typekey has a keyfilter for the DoctorCategoryType typekey.

Later lessons discuss how to configure the user interface with Page Configuration Files and the Input widgets.

## Lesson objectives review

- You should now be able to:
  - Describe the functionality of a typelist
  - Create a typelist and typelist extension
  - Create a typekey field
  - Create a typekey field that references a typelist filter

## Review questions

1. Describe some of the differences between the files in the ...\\metadata\\typelist\\ and ...\\extensions\\typelist\\ folders.
2. Which is a valid code value for a typecode in a food coloring typelist?
  - a) RedDye#1
  - b) YellowDye40
  - c) OrangeDye:10
3. Which typecode attribute determines the order in which typecodes are listed in the user interface?
4. What is the key difference between a typekey and a typekey with a defined typefilter attribute?

## Answers

- 1) Files under metadata are read-only are specific to the base configuration. You cannot create typelist files in the metadata directory but can create typelist files in the extensions directory. Only internal typelist extension files (TIX) are in the metedata directory. Only external typelist files (TTX) are in the extensions directory. Both directories contain typelist (TTI) files.
- 2) The typecode must be unique within the typelist and must have 50 characters or fewer. The value for the code must be an alphanumeric value.
  - 2a) Not valid
  - 2b) Valid
  - 2c) Not valid
- 3) The priority attribute determines the sort order in the User Interface for a typelist. After priority, the name (not code) determines the sort order. Entries with priority -1 appear at the end of the list
- 4) A typekey is only limited by the values in the typelist. A typekey with a defined typefilter attribute is limited to the named subset of values as defined by the typekey filter in the typelist.

# Notices

**Copyright © 2001-2014 Guidewire Software, Inc. All rights reserved.**

Guidewire, Guidewire Software, Guidewire ClaimCenter, Guidewire PolicyCenter, Guidewire BillingCenter, Guidewire Reinsurance Management, Guidewire ContactManager, Guidewire Vendor Data Management, Guidewire Client Data Management, Guidewire Rating Management, Guidewire InsuranceSuite, Guidewire ContactCenter, Guidewire Studio, Guidewire Product Designer, Guidewire Live, Guidewire DataHub, Guidewire InfoCenter, Guidewire Standard Reporting, Guidewire ExampleCenter, Guidewire Account Manager Portal, Guidewire Claim Portal, Guidewire Policyholder Portal, ClaimCenter, BillingCenter, PolicyCenter, InsuranceSuite, Gosu, Deliver Insurance Your Way, and the Guidewire logo are trademarks, service marks, or registered trademarks of Guidewire Software, Inc. in the United States and/or other countries.

All other trademarks are the property of their respective owners.

**This material is confidential and proprietary to Guidewire and subject to the confidentiality terms in the applicable license agreement and/or separate nondisclosure agreement.**

This file and the contents herein are the property of Guidewire Software, Inc. Use of this course material is restricted to students officially registered in this specific Guidewire-instructed course, or for other use expressly authorized by Guidewire. Replication or distribution of this course material in electronic, paper, or other format is prohibited without express permission from Guidewire.

Guidewire products are protected by one or more United States patents.

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

42

G U I D E W I R E