Harmless Changes: “Change is coming”

Overview

– Problem description
– What are harmless changes?
  - Allowed harmless changes per datamodel item
– Added support for harmless changes
  - Control Centre
  - During install of a survey
– Behind the scenes: data conversion
  - Conversion differs for each data partition type
Problem description

– You want to make simple changes to a survey that is already in production
  - Change to a survey item often causes that the existing data becomes incompatible with the new datamodel
  - For example, if you apply a change like this:
– Survey must be replaced on the server
  - Survey must be taken offline
  - Existing data must be converted
  - Existing session data cannot be used anymore
– Harmless changes support should make migrating to a newer version more easier in many situations

Demo: incompatible data

What are harmless changes?

– Changes to a Blaise datamodel that do not affect already collected and/or existing data
– Goal:
  - Update surveys during production without the need to take them offline
    • Session data will not be lost
    • Survey data remains available
– Applies to:
  - Updates to surveys installed on a server
  - Updates to surveys installed in the apps
What are harmless changes?

The general rules are:
- No (relevant) items may be removed
- Items can be added
  - Fields, but also blocks
  - Checks
- Each item of the new collection must be a harmless extension of the related item (i.e. with the same name) of the old collection

Datamodel level

The following items can be harmless extended:
- Modes
- Keys
- (non-Aux) Fields
- Checks
- Field Properties

In case of multi-mode data models
- new mode-specific data model must be a harmless extension of the old mode-specific data model
Keys

- The key kind must be identical
- The Fields must be key-identical:
  - The order and the count of the fields is the same
  - Each key-field of the new key-field must be a harmless extension of the related old key-field that preserves the (maximum) length of the value

Fields

- **The following items can be harmless extended**
  - Type
  - The Allowed Special Answers (for data fields)
- **Restrictions**
  - The FieldKind (AuxField or Field) must be the same
  - For each AuxField in the new data model there may not be a datafield (with the same name) in the old datamodel
Types

- **Type Structure must be identical**
  - If a field is a block in the old model, then it must be a block in the new model, string must be string, et cetera

- **Additional requirements per Type**:
  - **Block**
    - The following items can be harmless extended
      - (non-Aux) Fields
      - Checks
  - **Array**
    - The following items can be harmless extended
      - Index type
      - Member type
      - Lowerbound index <= Old lowerbound
      - Upperbound index >= Old upperbound

---

Types

- **Classification**
  - The following items can be harmless extended:
    - Levels
    - Required Depth (may not be more restrictive)
    - Items (similar to Enumeration)

- **Enumeration**
  - Items can be extended
  - All codes of the old type must be in the new type
  - For each category in the new type there may not be a category (with the same name) that has a different code
<table>
<thead>
<tr>
<th>Types</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>– Integer</strong></td>
</tr>
<tr>
<td>- New minimum value &lt;= Old minimum value</td>
</tr>
<tr>
<td>- New maximum value &gt;= Old maximum value</td>
</tr>
<tr>
<td><strong>– Real</strong></td>
</tr>
<tr>
<td>- New number of decimals &gt;= Old number of decimals</td>
</tr>
<tr>
<td>- New minimum value &lt;= Old minimum value</td>
</tr>
<tr>
<td>- New maximum value &gt;= Old maximum value</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Types</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>– Set</strong></td>
</tr>
<tr>
<td>- The new cardinality must be greater or equal than the old cardinality</td>
</tr>
<tr>
<td>- The new Member type must be a harmless extension of the old Member type.</td>
</tr>
<tr>
<td><strong>– String/Open</strong></td>
</tr>
<tr>
<td>- New maximum length &gt;= Old maximum length</td>
</tr>
<tr>
<td>- If the new type has a restriction (regular expression), the old type should have the same restriction</td>
</tr>
</tbody>
</table>
Checks and FieldProperties

- **Check**
  - The derived Type (Boolean, or Array) of the new check must be a harmless extension of the derived old check

- **FieldProperty**
  - The type of the new FieldProperty must be a harmless extension of the type of the old FieldProperty

Support in Control Centre

- Options -> Harmless changes support
  - AutoUpdate
  - Set to AutoUpdate when BDIX is valid and compatible with meta

- Tries to keep the bdix and its data compatible with datamodel changes

- Changes BDIX and converts existing data
  - Make sure that your project’s bdix doesn’t point to production data!!!

Demo: added support in Control Centre
Support during install

- Installer ‘sees’ whether a datamodel is a harmless extension
- Options to be selected during install
  - Overwrite existing data = No
  - Harmless update = Yes
- If harmless:
  - No need to reinstall survey
  - Session data will be preserved
  - Bdix will be updated
  - Existing data will be converted automatically
- Selected options during install apply to the apps as well

Demo: added support during install

Behind the scenes: data conversion

- BDIX
  - DataChecksums are updated
  - Changes are made to table definitions if needed
- Data
  - How depends on data partition type
    • Stream
      • Data checksums in streams are updated
    • Flat Blocks and Flat, No Blocks
      • Columns will be added for new fields
      • Existing columns will be extended to match the new field definition
    • In Depth
      • FieldIds and CheckIds will be updated

Demo: conversion data partition types
Rollback mechanism

- Multiple steps are executed during installation
  - New meta files are copied to survey’s deploy folder
  - Bdix is altered
  - In target database: table structures might be altered / fieldids might be updated
- It is important to have a rollback mechanism in place
- Data Server
  - Previous meta and data files are copied to backup folder
    • Are copied back to survey’s installation folder if harmless update fails
  - Data updates in databases are executed within transaction
- Changes are roll backed entirely if something goes wrong

Data conversion and server databases

- Applies to MySQL, SQL, Oracle, DB2, PostgreSQL
  - In depth, Flat block and Flat no blocks require Alter Table and Create Table SQL statements
  - Make sure that the user in ConnectionString has enough rights to execute these statements
- Microsoft Access: only stream is supported
  - Update SQL statement is not compatible