Software-ontwerp: Tablr Iteration 2

QUINTEN BRUYNSERAEDE

MARTIJN SLAETS

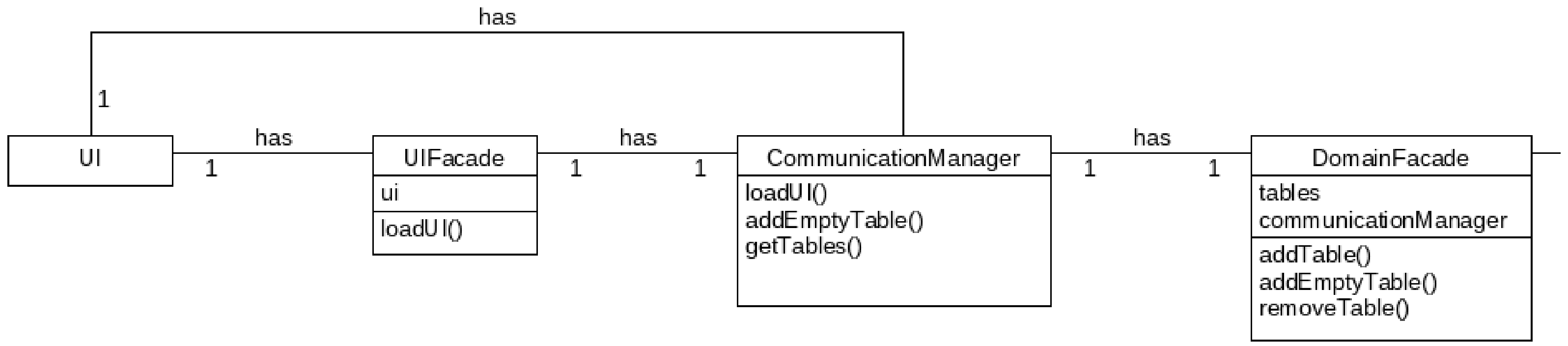
TOM DE BACKER

IGNACE BLEUKX

1. Design: then vs Now

Iteration 1:

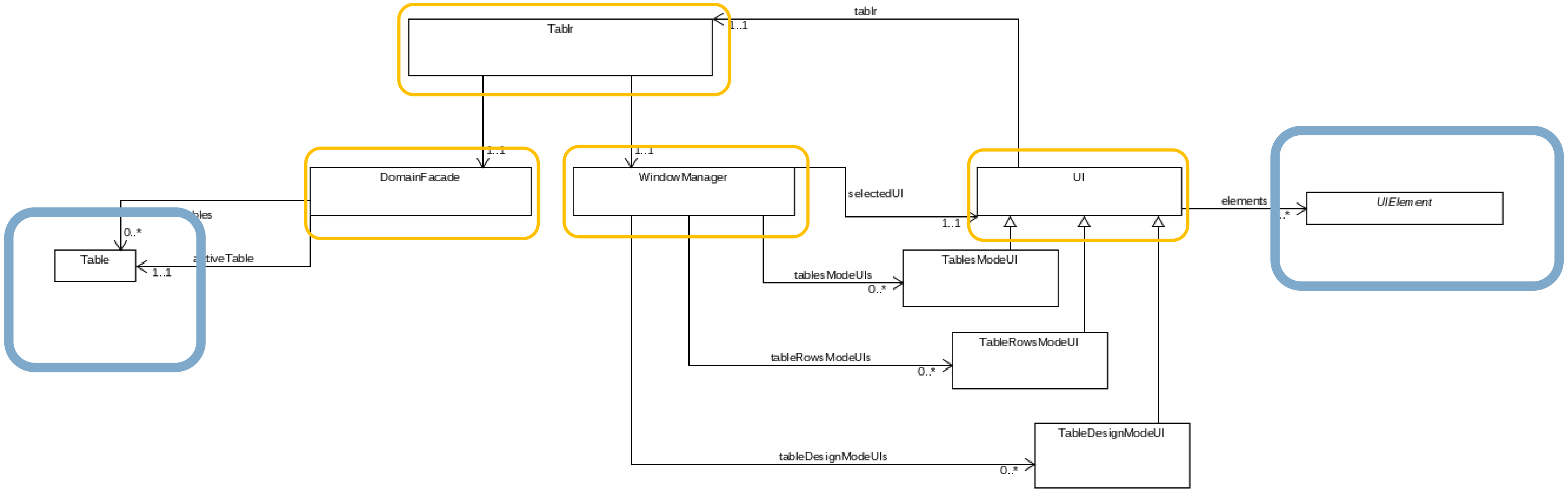
- CommunicationManager was used by <u>every object</u> to modify Domain or UI
- CommunicationManager made use of UIFacade and DomainFacade



1. Design: then vs Now

Iteration 2:

- All DomainElements and UIElements are free of references to 'Tablr'-specific Classes



1. Design: then vs Now

Iteration 1:

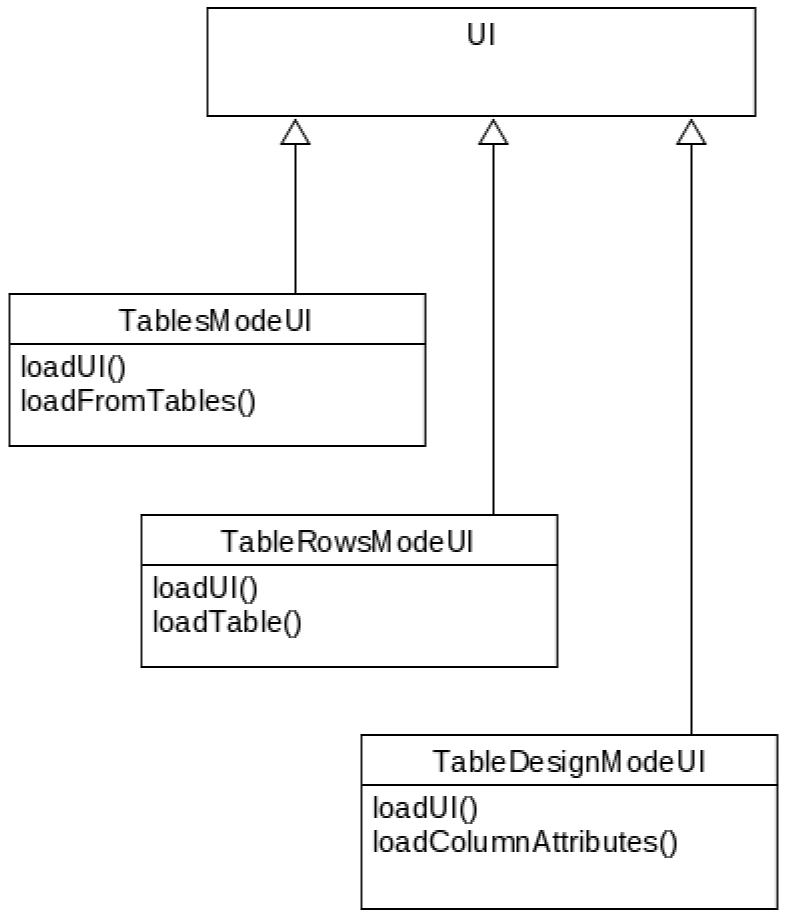
The loading of different UI's was handled in different UIElements

- e.g. ListView.loadFromTables()

1. Design: then vs Now

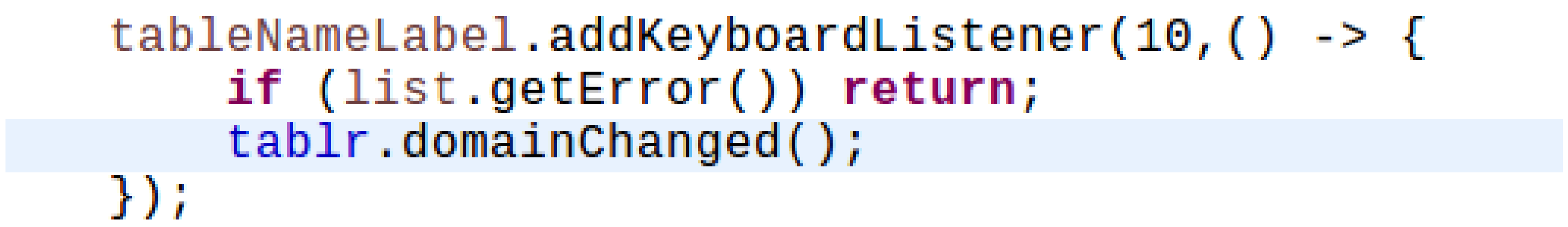
Iteration 2:

- Subwindows inherit from superclass UI
- Method loadUI() to create the necessary components
- All Tablr-logic is specified in the loading of a UI, not in UIElements



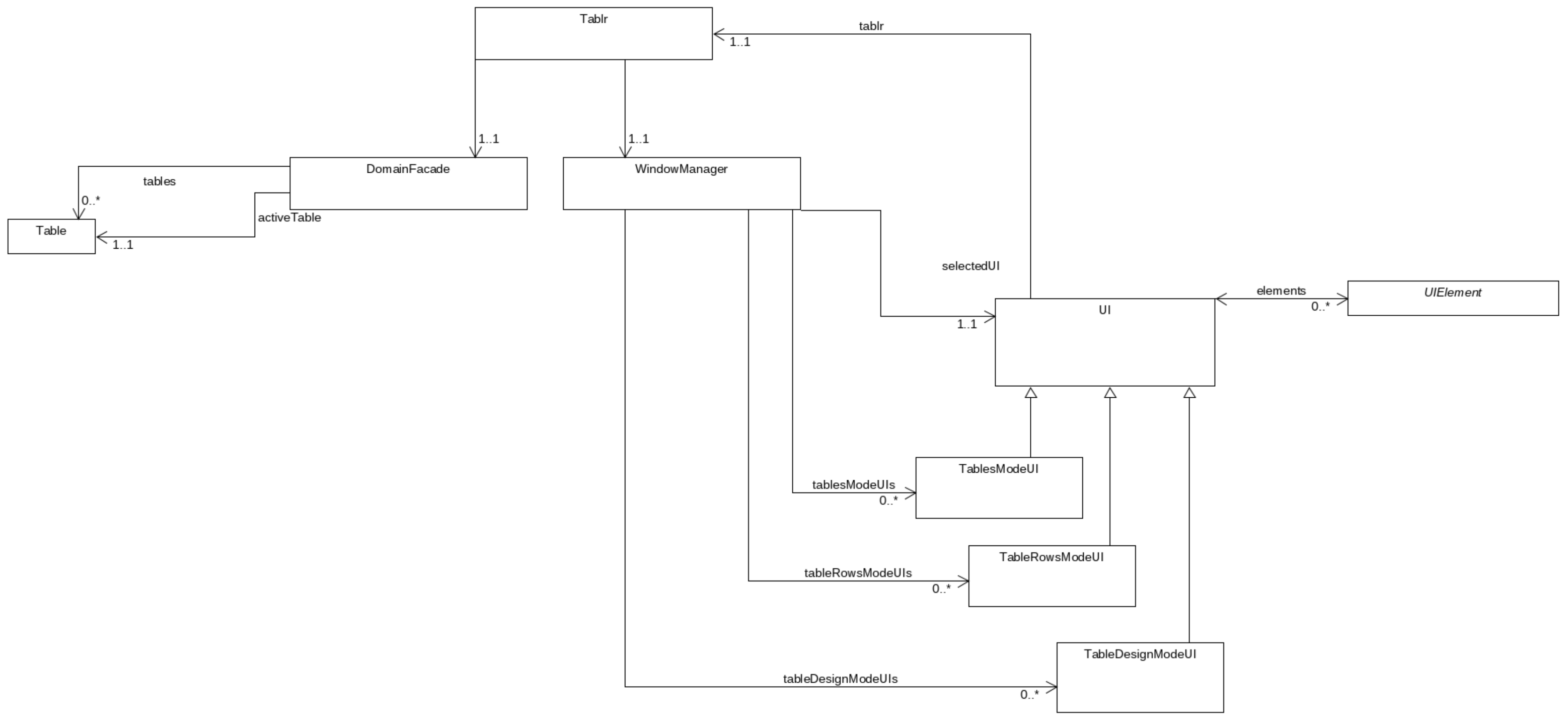
1. Design: then vs Now

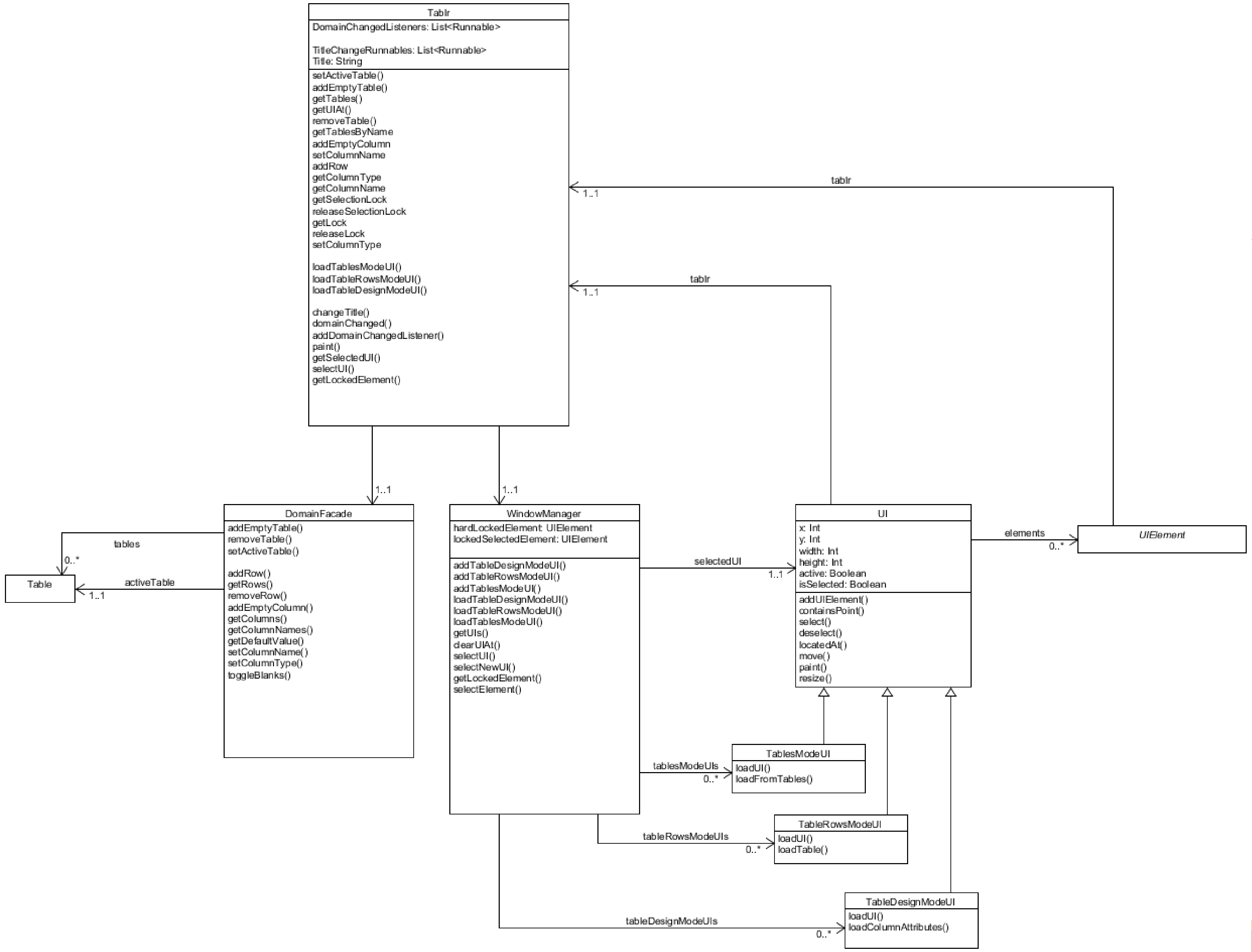
Iteration 2: extensive use of Listeners to specify behaviour of UIElements



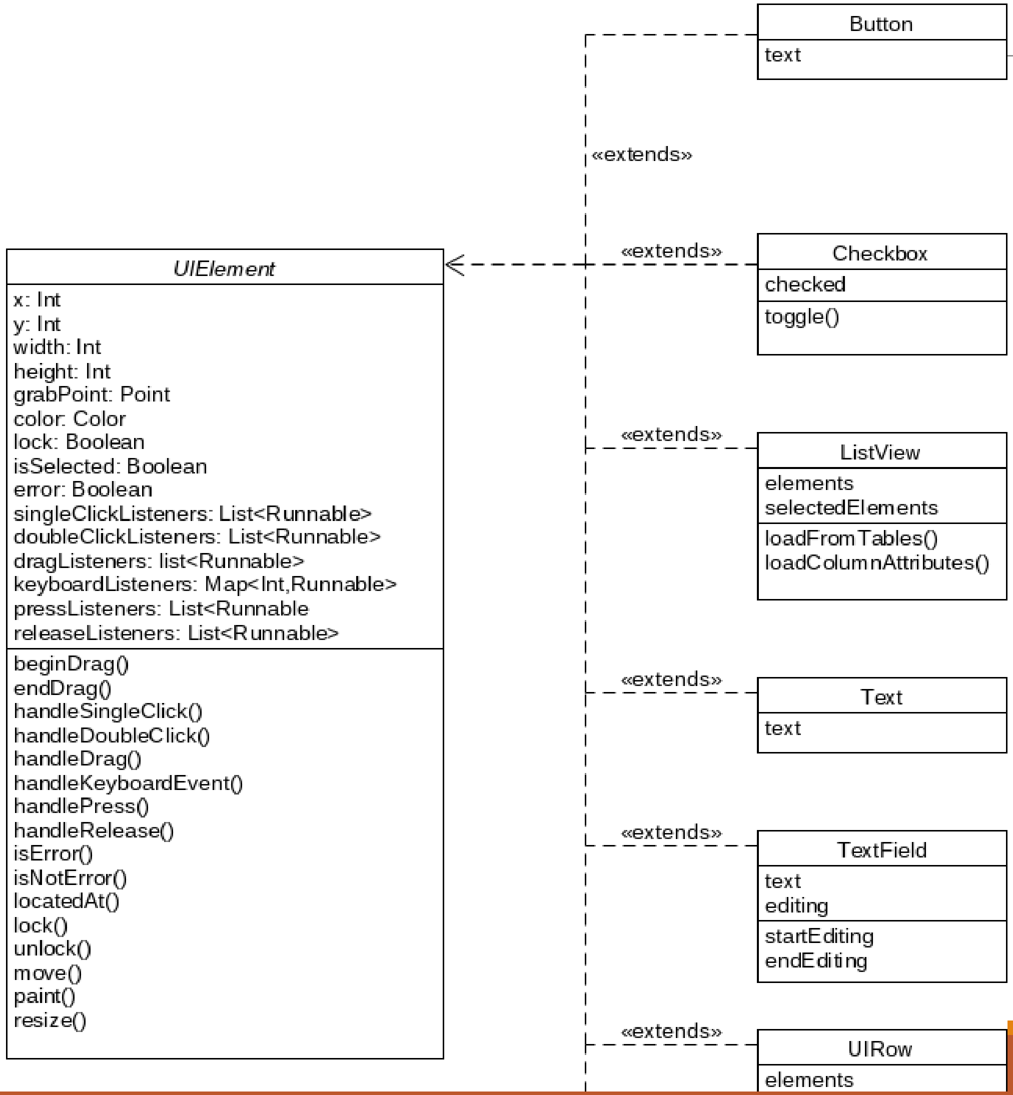
```
tableNameLabel.addKeyboardListener(10,() -> {
    if (list.getError()) return;
    tablr.domainChanged();
});
```

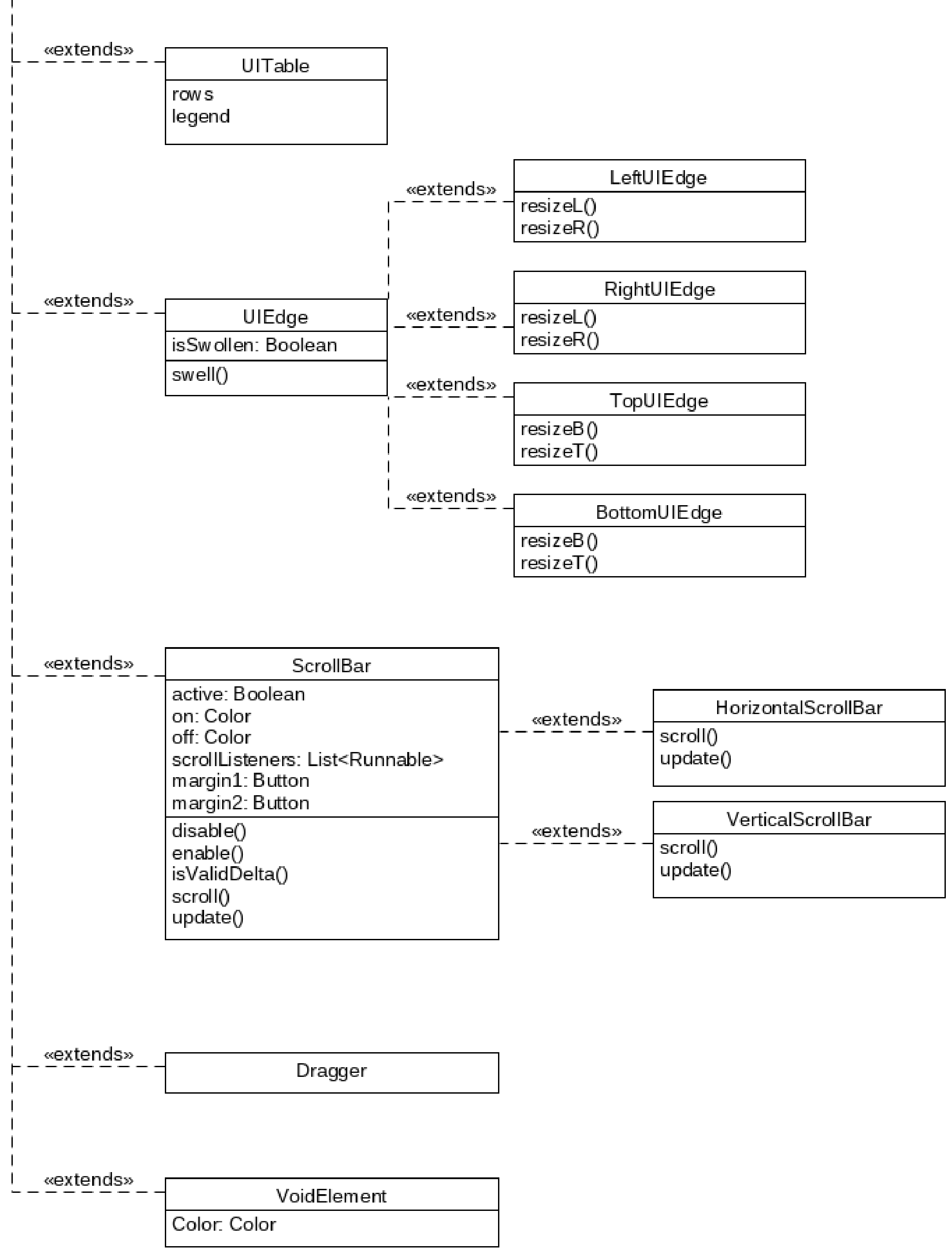
1. Design: class diagram



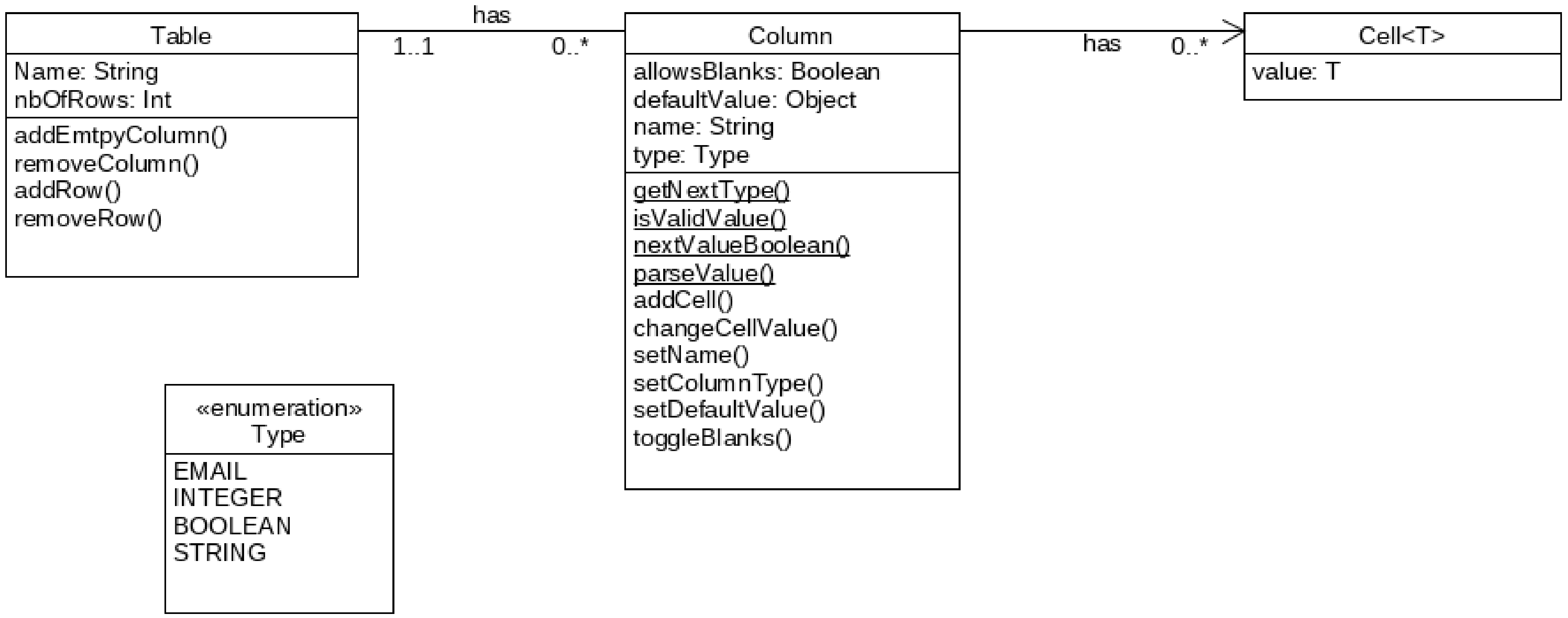

Tablr
DomainChangedListeners: List<Runnable>
TitleChangeRunnables: List<Runnable>
Title: String
setActiveTable()
addEmptyTable()
getTables()
getUIAt()
removeTable()
getTablesByName
addEmptyColumn
setColumnName
addRow
getColumnType
getColumnName
getSelectionLock
releaseSelectionLock
getLock
releaseLock
setColumnType
loadTablesModeUI()
loadTableRowsModeUI()
loadTableDesignModeUI()
changeTitle()
domainChanged()
addDomainChangedListener()
paint()
getSelectedUI()
selectUI()
getLockedElement()
tablr
1..1
tablr
1..1
1..1
1..1
DomainFacade
addEmptyTable()
removeTable()
setActiveTable()
addRow()
getRows()
removeRow()
addEmptyColumn()
getColumns()
getColumnNames()
getDefaultValue()
setColumnName()
setColumnType()
toggleBlanks()
tables
0..*
Table
activeTable
1..1
WindowManager
hardLockedElement: UIElement
lockedSelectedElement: UIElement
addTableDesignModeUI()
addTableRowsModeUI()
addTablesModeUI()
loadTableDesignModeUI()
loadTableRowsModeUI()
loadTablesModeUI()
getUIs()
clearUIAt()
selectUI()
selectNewUI()
getLockedElement()
selectElement()
selectedUI
1..1
UI
x: Int
y: Int
width: Int
height: Int
active: Boolean
isSelected: Boolean
addUIElement()
containsPoint()
select()
deselect()
locatedAt()
move()
paint()
resize()
elements
0..*
UIElement
tablesModeUIs
0..*
TablesModeUI
loadUI()
loadFromTables()
tableRowsModeUIs
0..*
TableRowsModeUI
loadUI()
loadTable()
tableDesignModeUIs
0..*
TableDesignModeUI
loadUI()
loadColumnAttributes()

1. Design: UI elements



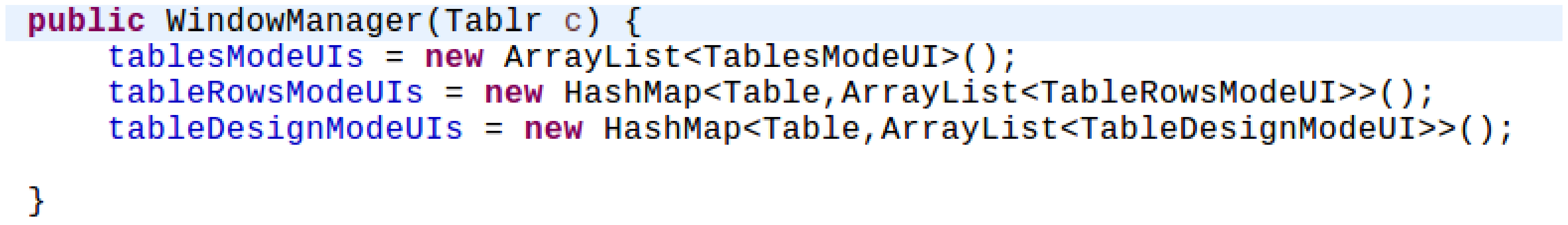

«extends»
UITable
rows
legend
«extends»
UIEdge
isSwollen: Boolean
swell()
«extends»
LeftUIEdge
resizeL()
resizeR()
«extends»
RightUIEdge
resizeL()
resizeR()
«extends»
TopUIEdge
resizeB()
resizeT()
«extends»
BottomUIEdge
resizeB()
resizeT()
«extends»
ScrollBar
active: Boolean
on: Color
off: Color
scrollListeners: List<Runnable>
margin1: Button
margin2: Button
disable()
enable()
isValidDelta()
scroll()
update()
«extends»
HorizontalScrollBar
scroll()
update()
«extends»
VerticalScrollBar
scroll()
update()
«extends»
Dragger
«extends»
VoidElement
Color: Color

1. Design: Domain



1. Design: handling subwindows

- WindowManager holds all UI's/Subwindows

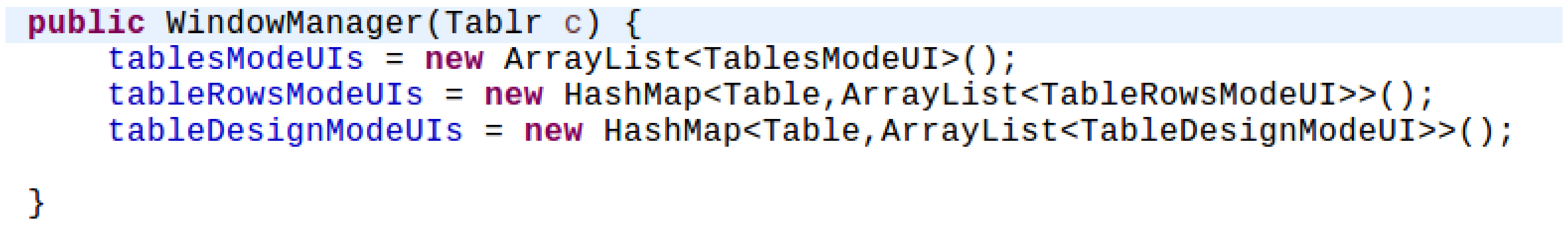


```java
public WindowManager(Tablr c) {
    tablesModeUIs = new ArrayList<TablesModeUI>();
    tableRowsModeUIs = new HashMap<Table,ArrayList<TableRowsModeUI>>();
    tableDesignModeUIs = new HashMap<Table,ArrayList<TableDesignModeUI>>();

}
```

1. Design: handling subwindows

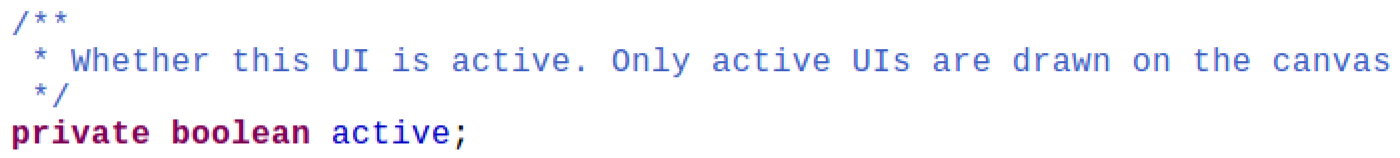
- WindowManager holds all UI's/Subwindows



```
public WindowManager(Tablr c) {
    tablesModeUIs = new ArrayList<TablesModeUI>();
    tableRowsModeUIs = new HashMap<Table,ArrayList<TableRowsModeUI>>();
    tableDesignModeUIs = new HashMap<Table,ArrayList<TableDesignModeUI>>();

}
```

- Opening/Closing means activating/deactivating



```
/**
 * Whether this UI is active. Only active UIs are drawn on the canvas
 */
private boolean active;
```

1. Design: handling subwindows

- Opening a subwindow multiple times means cloning an existing UI at different coordinates

```
@Override
public UI clone(){
    UI clone = new UI(getX(),getY(),getWidth(),getHeight());
    ArrayList<UIElement> clonedElements = new ArrayList<UIElement>();
    elements.stream().forEach(e -> clonedElements.add(e.clone()));
    clone.elements = clonedElements;
    return clone;
}
```

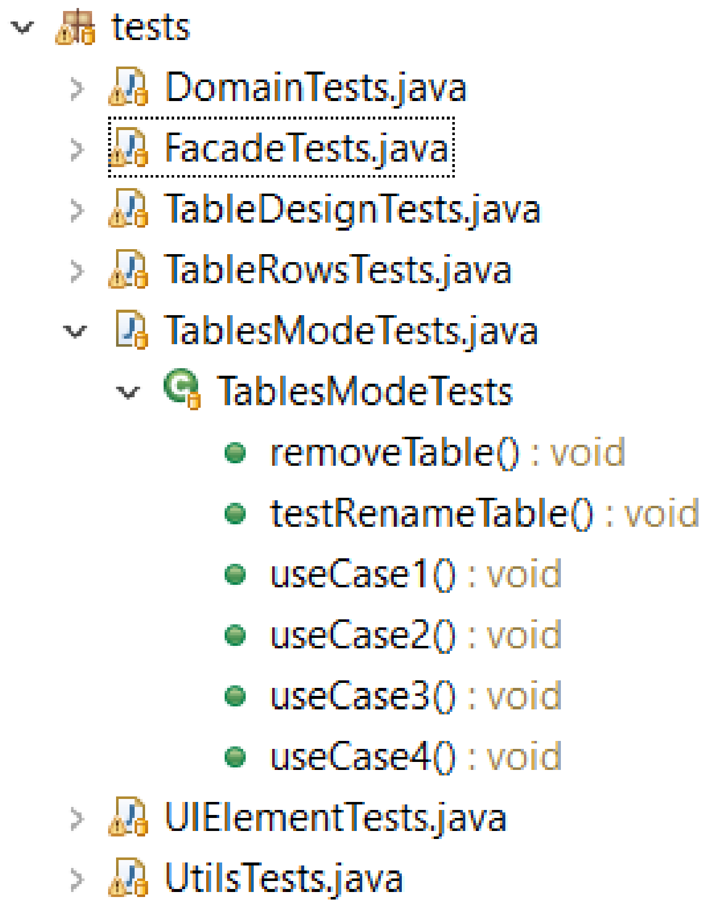
3. Extensibility

- All program logic is contained in the loadUI method of UI's, can be modified in one place.

- UI and Domain stand on their own, collect all actions in List<Runnable>. Additional concepts can always follow this logic.

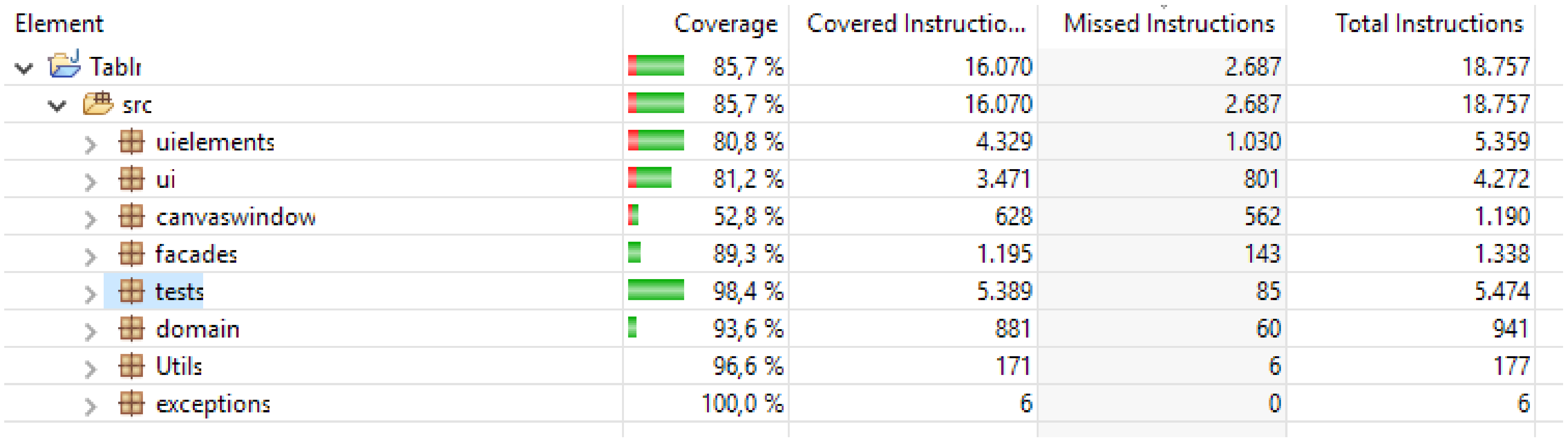
4. Testing Approach

- Start by testing all Use Cases (55% coverage)
- Improve coverage with focused testing of remaining parts



```
tests
  DomainTests.java
  FacadeTests.java
  TableDesignTests.java
  TableRowsTests.java
  TablesModeTests.java
    TablesModeTests
      removeTable() : void
      testRenameTable() : void
      useCase1() : void
      useCase2() : void
      useCase3() : void
      useCase4() : void
  UIElementTests.java
  UtilsTests.java
```

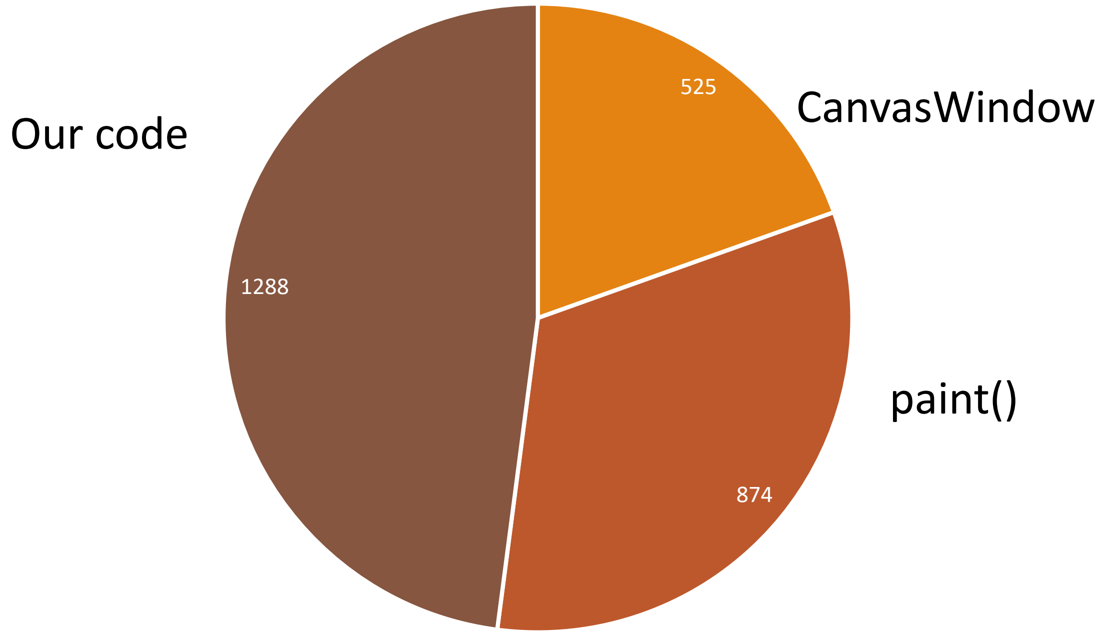
4. Testing Approach – Total Coverage



Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
Tablr	85,7 %	16.070	2.687	18.757
src	85,7 %	16.070	2.687	18.757
uielements	80,8 %	4.329	1.030	5.359
ui	81,2 %	3.471	801	4.272
canvaswindow	52,8 %	628	562	1.190
facades	89,3 %	1.195	143	1.338
tests	98,4 %	5.389	85	5.474
domain	93,6 %	881	60	941
Utils	96,6 %	171	6	177
exceptions	100,0 %	6	0	6

4. Testing Approach

Missed instructions: 2687



Coverage without CanvasWindow and paint(): 92.4%

Overview of project management

This iteration:

- Domain Coordinator: Martijn
- Testing Coordinator: Ignace
- Design Coordinator: Tom & Quinten

Next iteration:

- Domain Coordinator: Quinten
- Testing Coordinator: Martijn
- Design Coordinator: Ignace & Tom

Spent hours: Group work

Quinten Bruynseraede: ~ 15 hours

Ignace Bleukx: ~ 25 hours

Tom De Backer: ~ 25 hours

Martijn Slaets: ~ 15 hours

Individual work

Quinten Bruynseraede: ~ 25 hours

Ignace Bleukx: ~ 50 hours

Tom De Backer: ~ 35 hours

Martijn Slaets: ~ 30 hours

Study

Quinten Bruynseraede: ~ 2 hour

Ignace Bleukx: ~ 2 hour

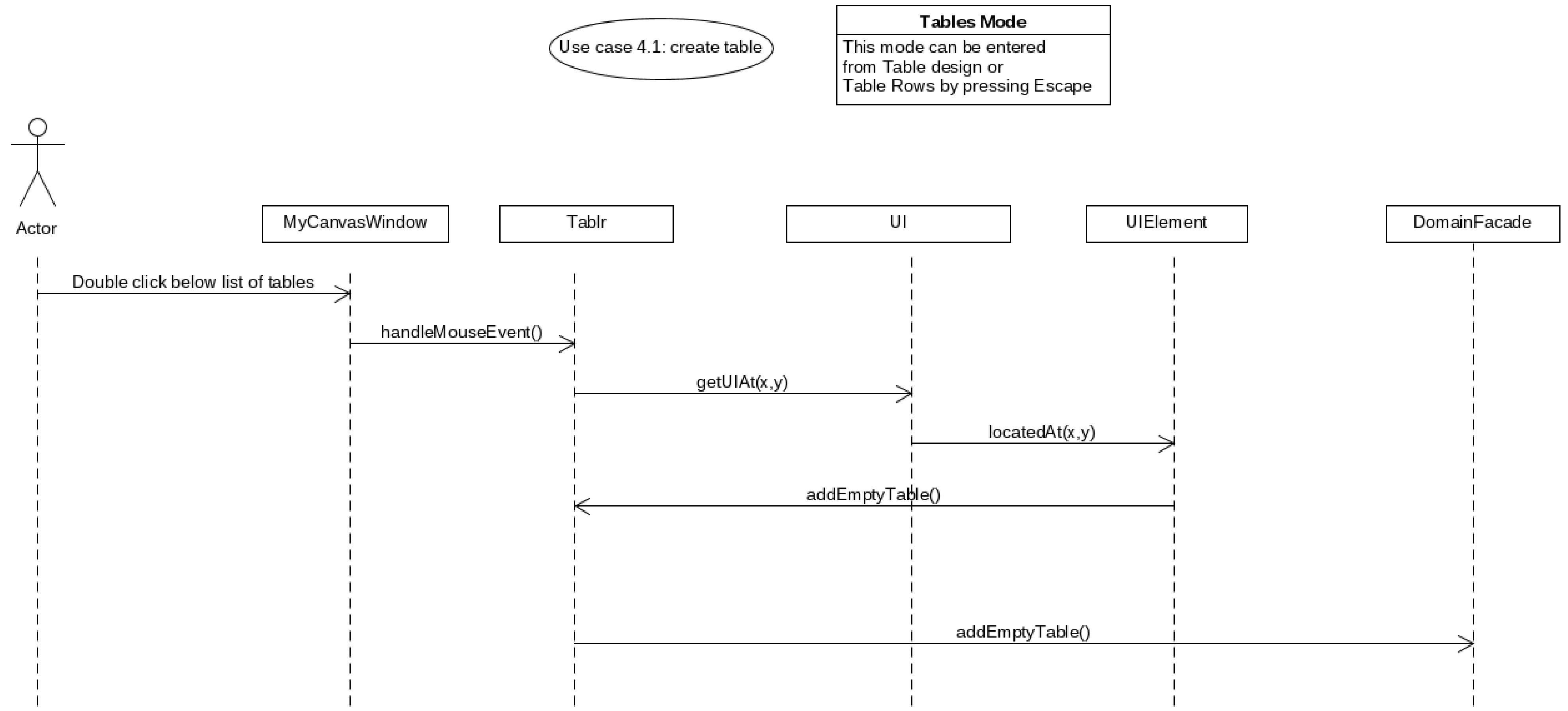
Tom De Backer: ~ 2 hour

Martijn Slaets: ~ 2 hour

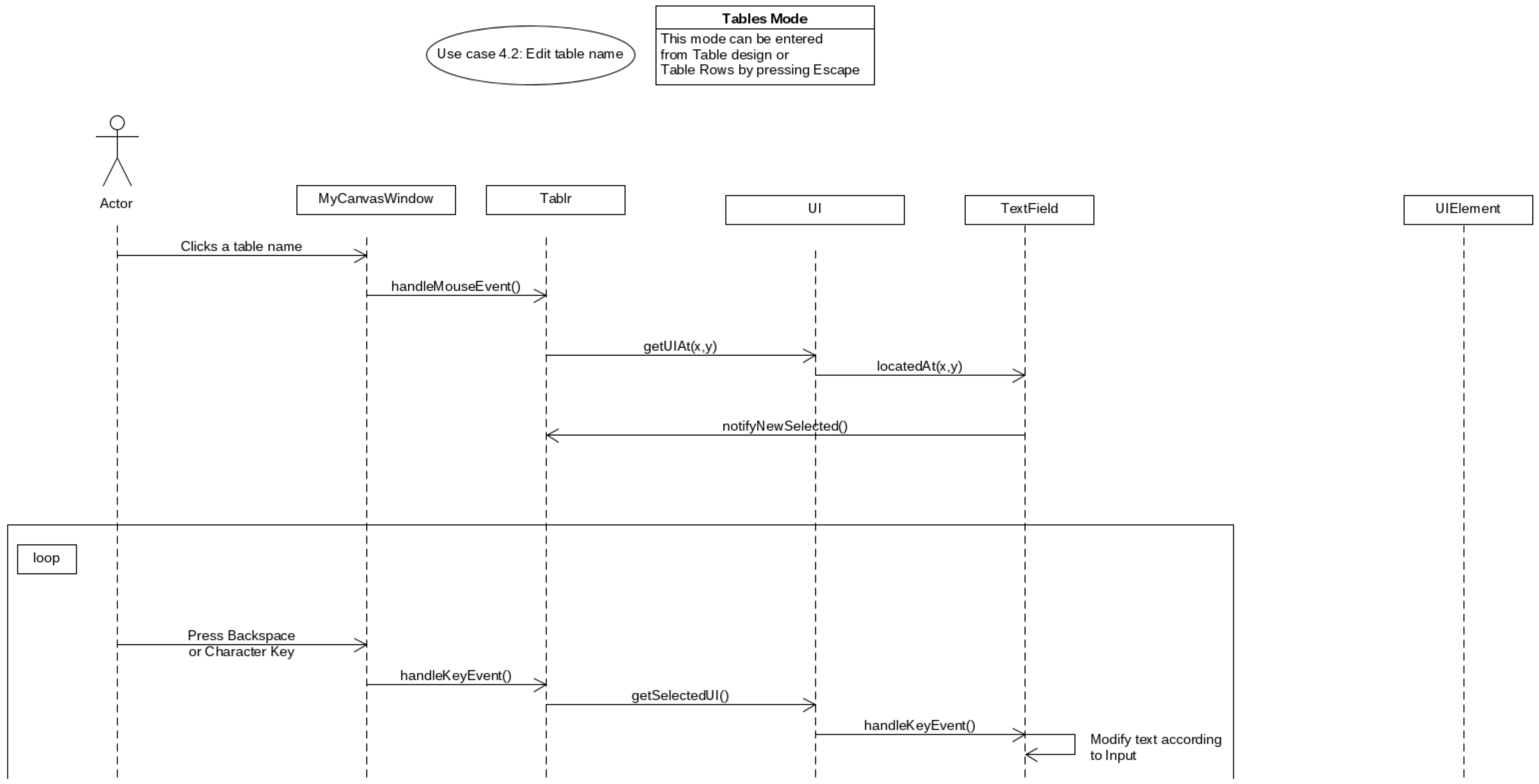
Use cases

1. Find UIElement that needs to act upon input
2. Invoke its singleClickHandler() / keyEventHandler()
   - Modifies UIElement
   - Uses a Tablr reference to modify Domain if necessary
3. Notify other UIElements if Domain changed

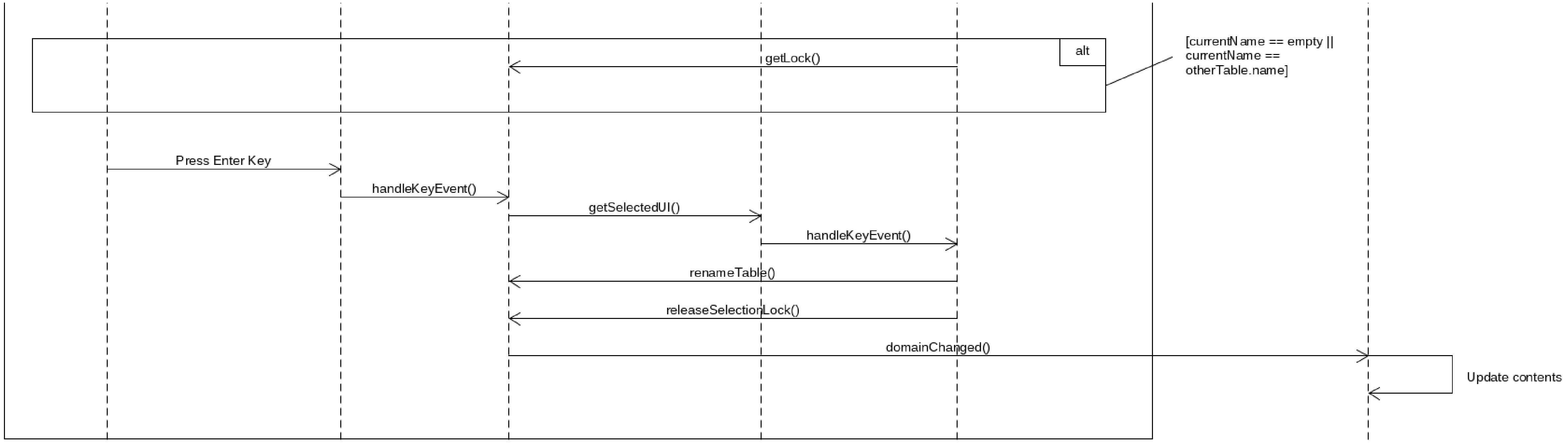
1. Adding a table


Use case 4.1: create table
Tables Mode
This mode can be entered from Table design or Table Rows by pressing Escape
Actor
MyCanvasWindow
Tablr
UI
UIElement
DomainFacade
Double click below list of tables
handleMouseEvent()
getUIAt(x,y)
locatedAt(x,y)
addEmptyTable()
addEmptyTable()

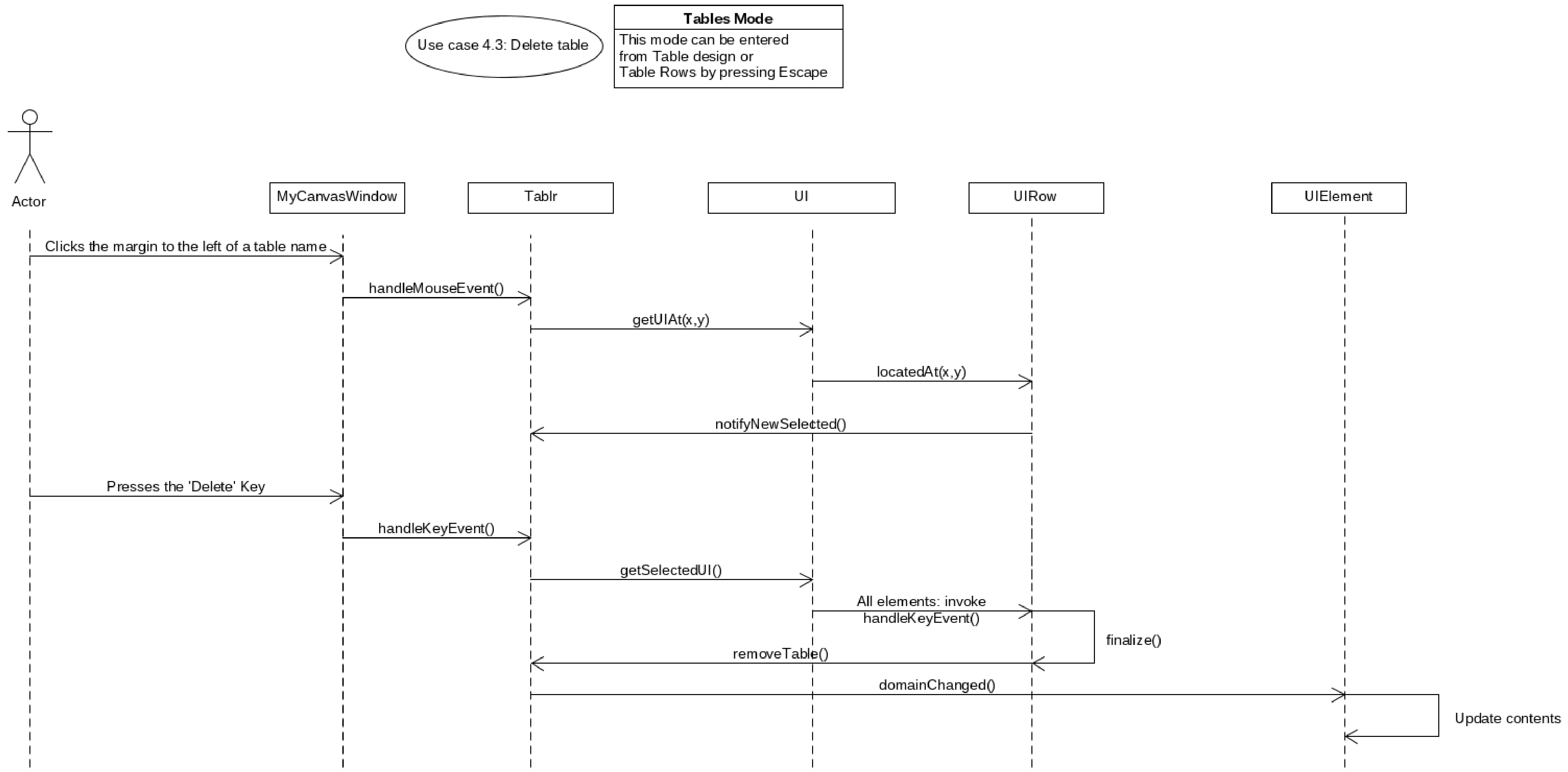
2. Edit a table name



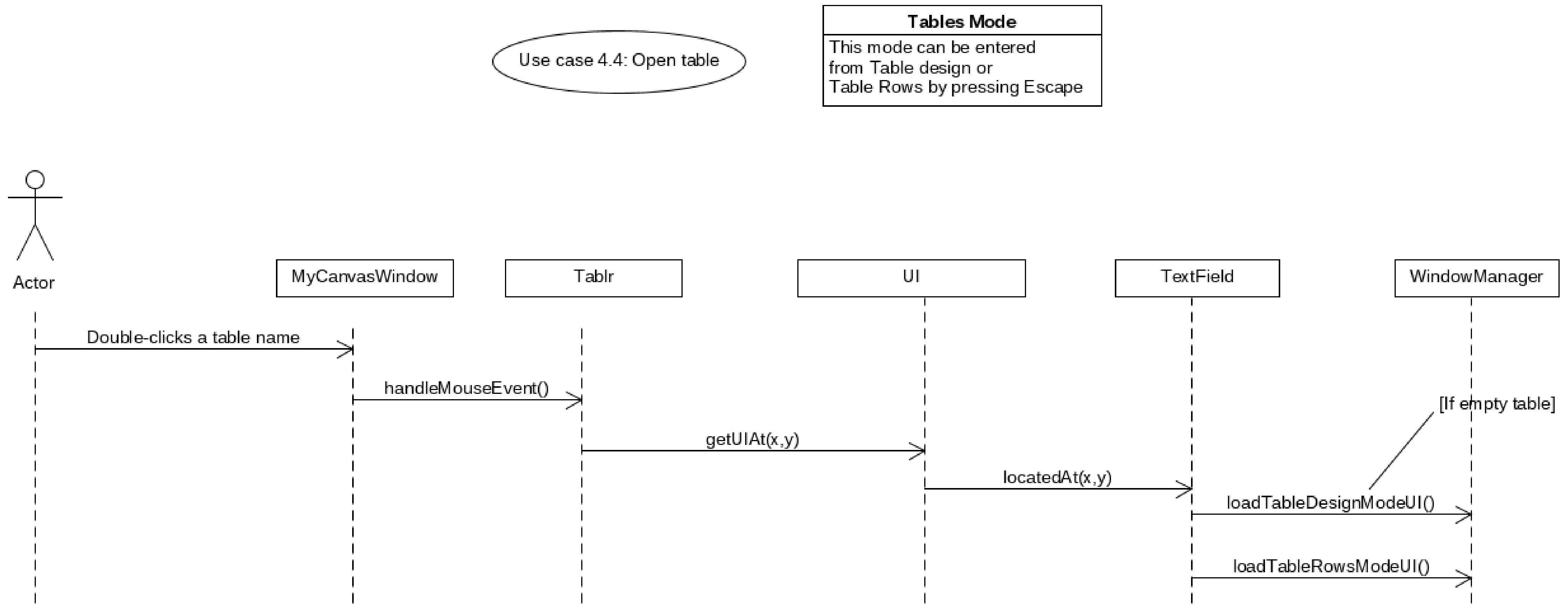
2. Edit a table name (continued)



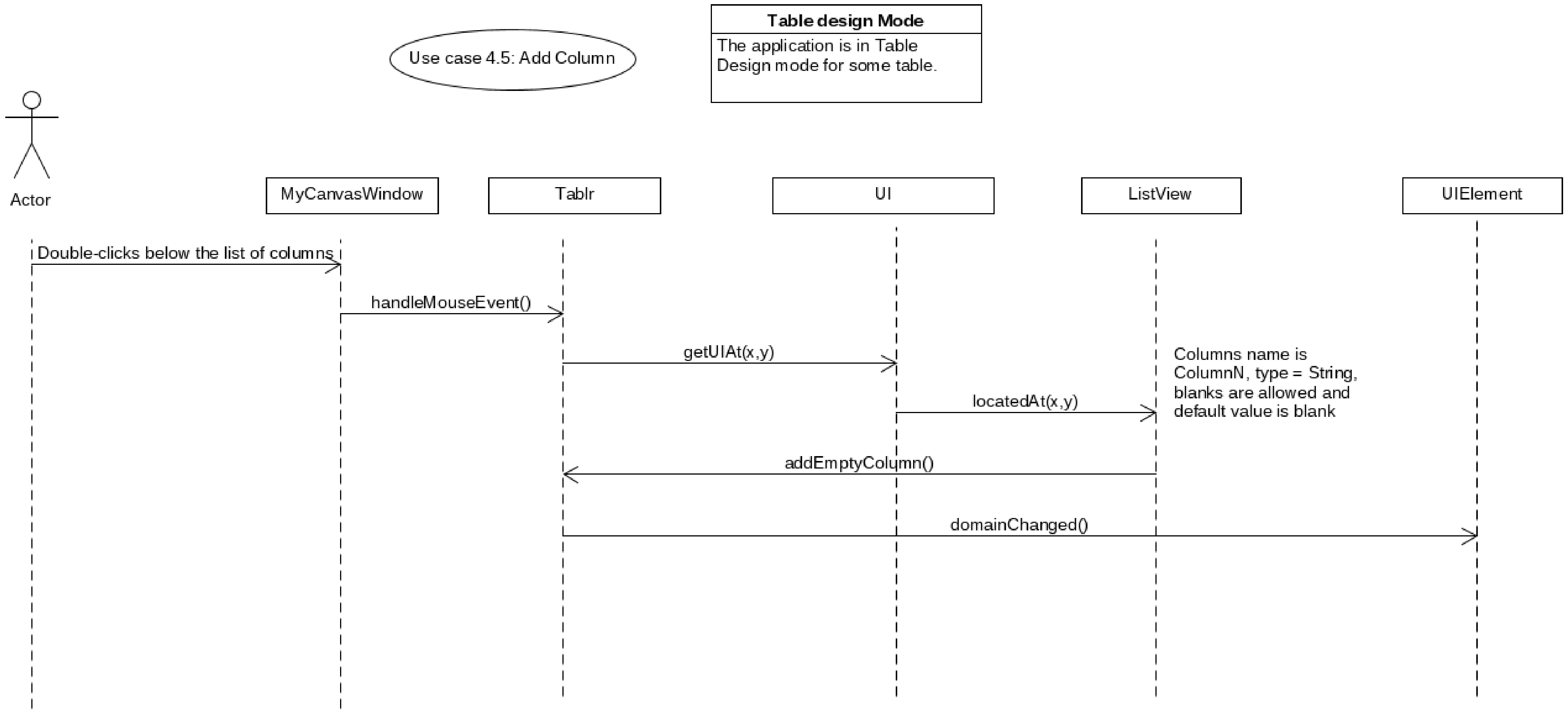
3. Delete table



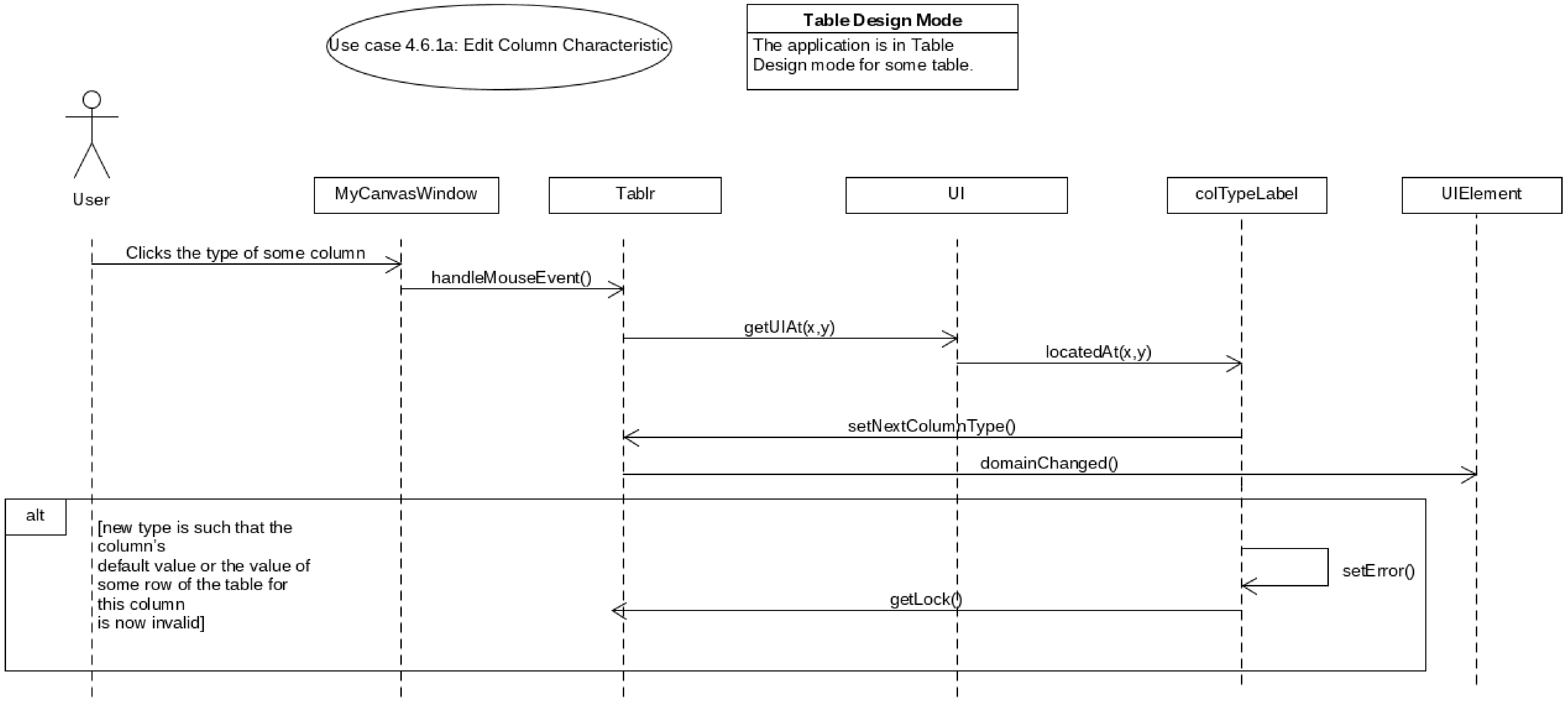
4. Open a Table



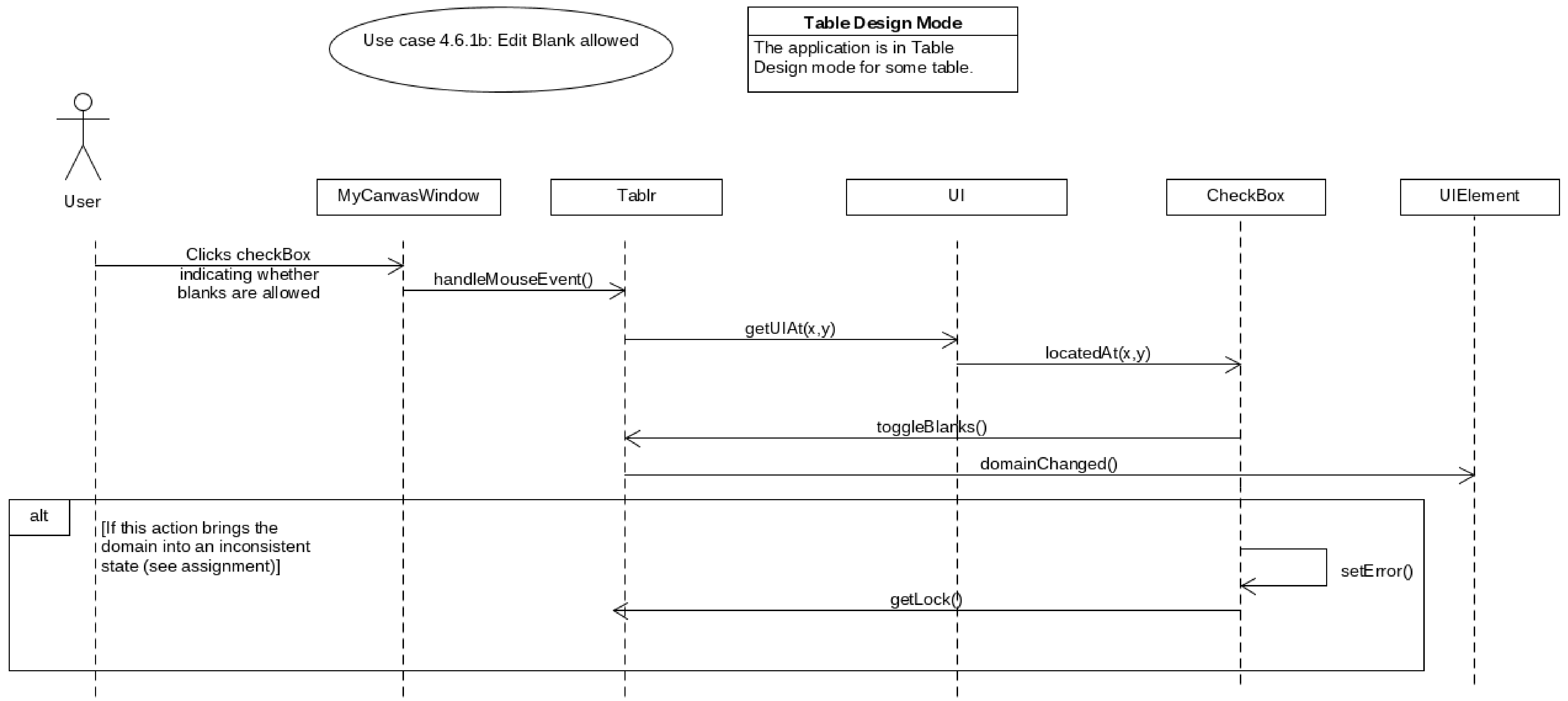
5. Add Column


Use case 4.5: Add Column
Table design Mode
The application is in Table Design mode for some table.
Actor
MyCanvasWindow
Tablr
UI
ListView
UIElement
Double-clicks below the list of columns
handleMouseEvent()
getUIAt(x,y)
locatedAt(x,y)
Columns name is ColumnN, type = String, blanks are allowed and default value is blank
addEmptyColumn()
domainChanged()

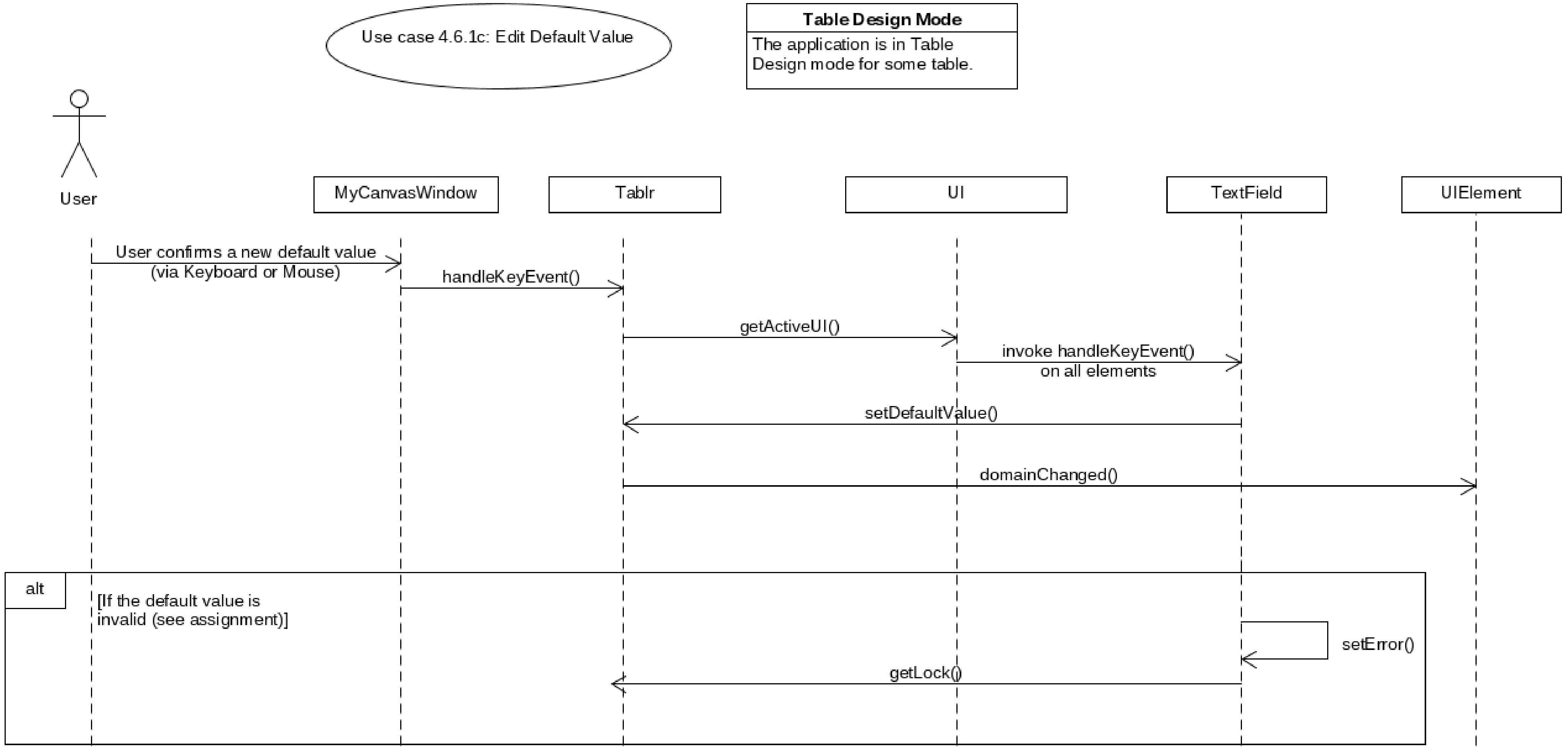
6. Edit Column Characteristic (a)


Use case 4.6.1a: Edit Column Characteristic
Table Design Mode
The application is in Table Design mode for some table.
User
MyCanvasWindow
Tablr
UI
colTypeLabel
UIElement
Clicks the type of some column
handleMouseEvent()
getUIAt(x,y)
locatedAt(x,y)
setNextColumnType()
domainChanged()
alt
[new type is such that the column's default value or the value of some row of the table for this column is now invalid]
setError()
getLock()

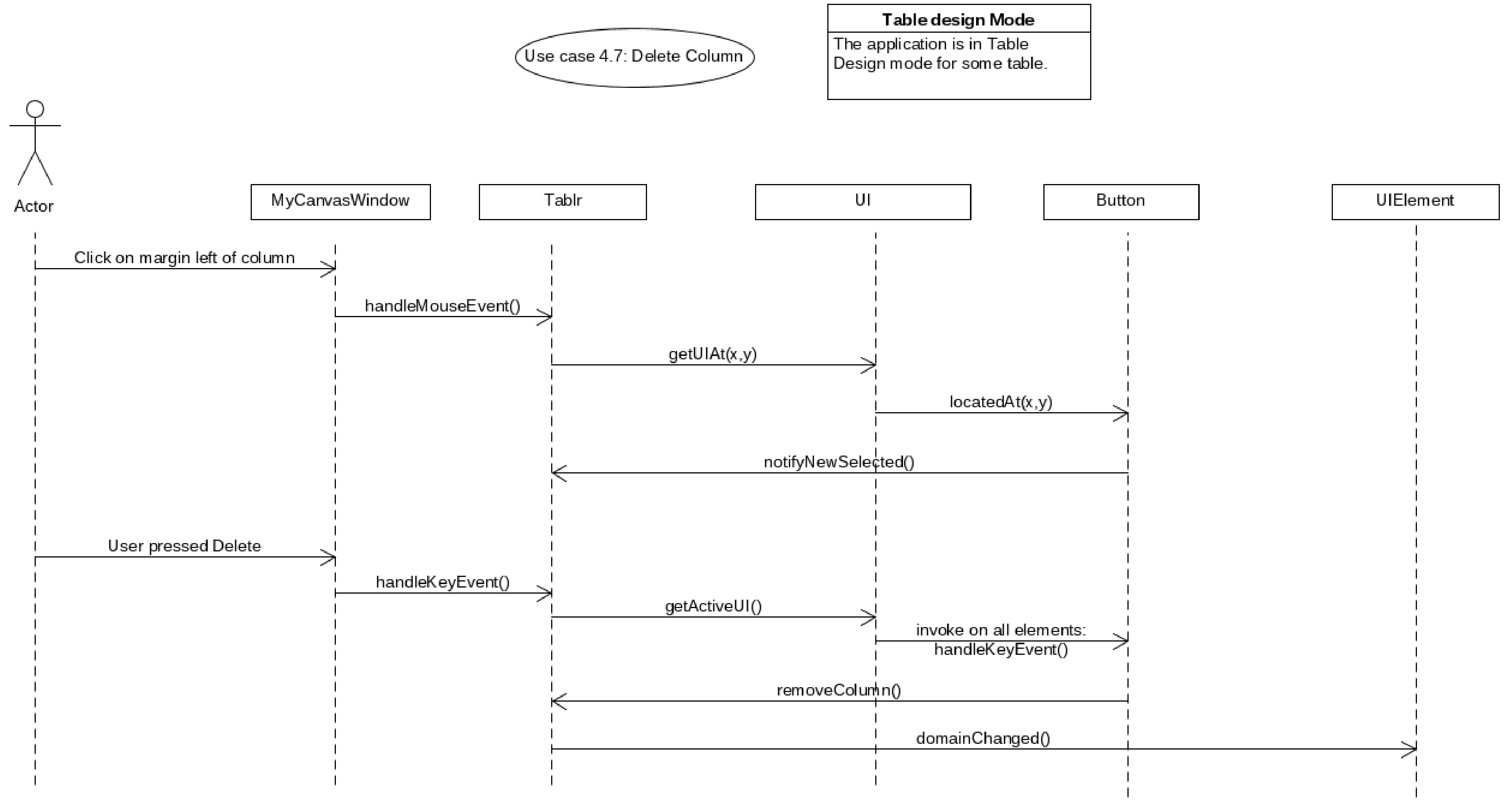
6. Edit Column Characteristic (b)


Use case 4.6.1b: Edit Blank allowed
Table Design Mode
The application is in Table Design mode for some table.
User
MyCanvasWindow
Tablr
UI
CheckBox
UIElement
Clicks checkBox indicating whether blanks are allowed
handleMouseEvent()
getUIAt(x,y)
locatedAt(x,y)
toggleBlanks()
domainChanged()
alt
[If this action brings the domain into an inconsistent state (see assignment)]
setError()
getLock()

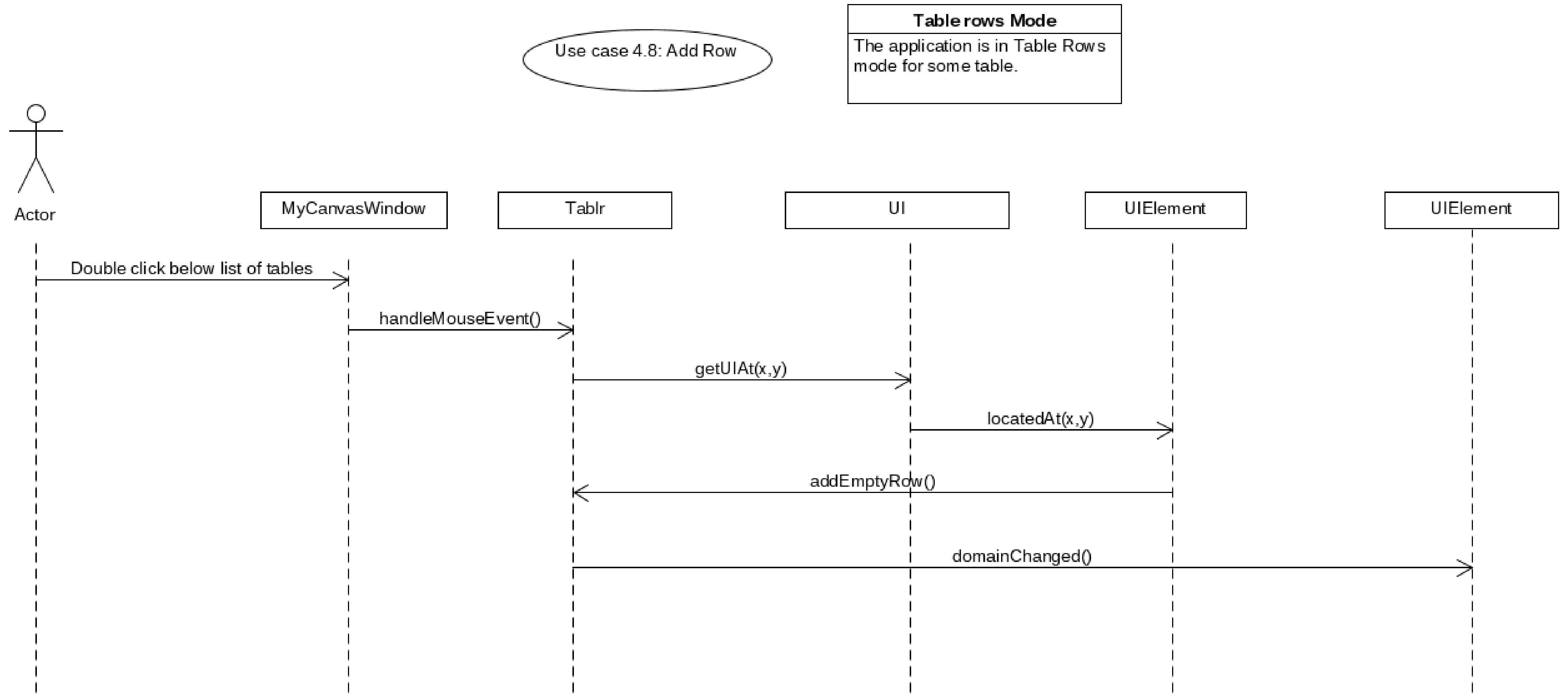
6. Edit Column Characteristic (c)


Use case 4.6.1c: Edit Default Value
Table Design Mode
The application is in Table Design mode for some table.
User
MyCanvasWindow
Tablr
UI
TextField
UIElement
User confirms a new default value (via Keyboard or Mouse)
handleKeyEvent()
getActiveUI()
invoke handleKeyEvent() on all elements
setDefaultValue()
domainChanged()
alt
[If the default value is invalid (see assignment)]
setError()
getLock()

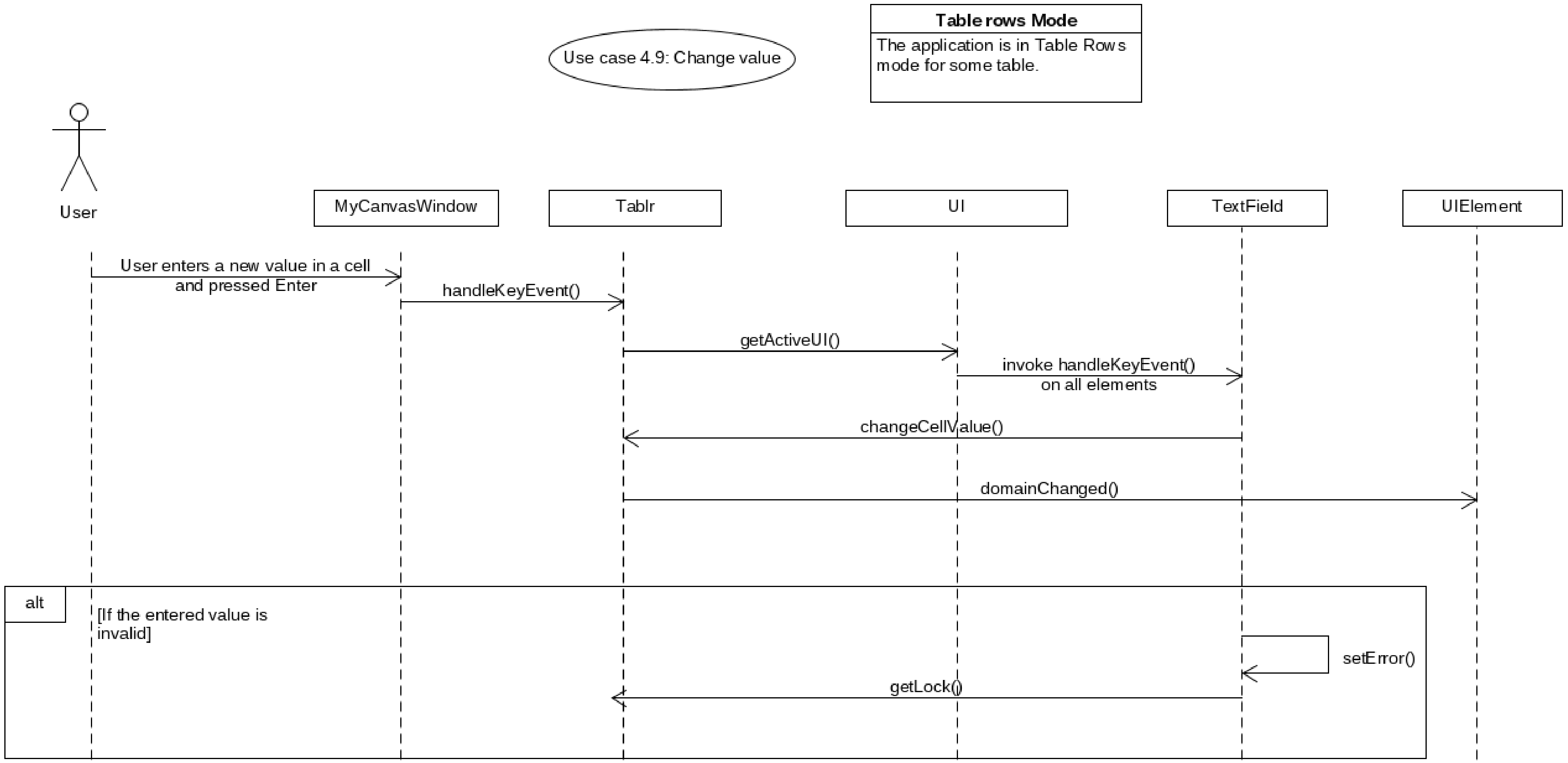
7. Delete Column



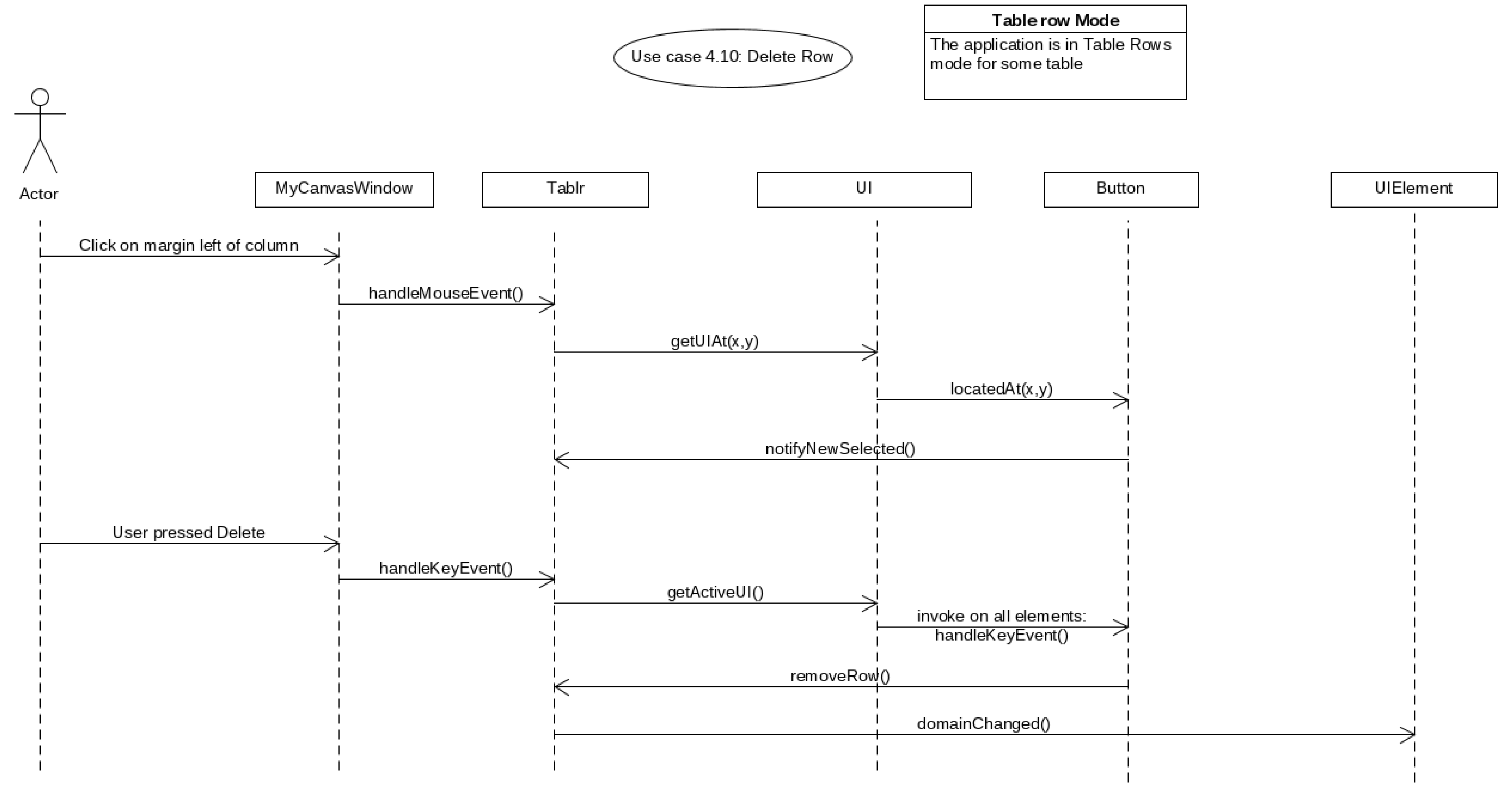
8. Add Row


Use case 4.8: Add Row
Table rows Mode
The application is in Table Rows mode for some table.
Actor
MyCanvasWindow
Tablr
UI
UIElement
UIElement
Double click below list of tables
handleMouseEvent()
getUIAt(x,y)
locatedAt(x,y)
addEmptyRow()
domainChanged()

9. Edit Row Value


Use case 4.9: Change value
Table rows Mode
The application is in Table Rows mode for some table.
User
MyCanvasWindow
Tablr
UI
TextField
UIElement
User enters a new value in a cell and pressed Enter
handleKeyEvent()
getActiveUI()
invoke handleKeyEvent() on all elements
changeCellValue()
domainChanged()
alt
[If the entered value is invalid]
setError()
getLock()

10. Delete Row


Use case 4.10: Delete Row
Table row Mode
The application is in Table Rows mode for some table
Actor
MyCanvasWindow
Tablr
UI
Button
UIElement
Click on margin left of column
handleMouseEvent()
getUIAt(x,y)
locatedAt(x,y)
notifyNewSelected()
User pressed Delete
handleKeyEvent()
getActiveUI()
invoke on all elements: handleKeyEvent()
removeRow()
domainChanged()