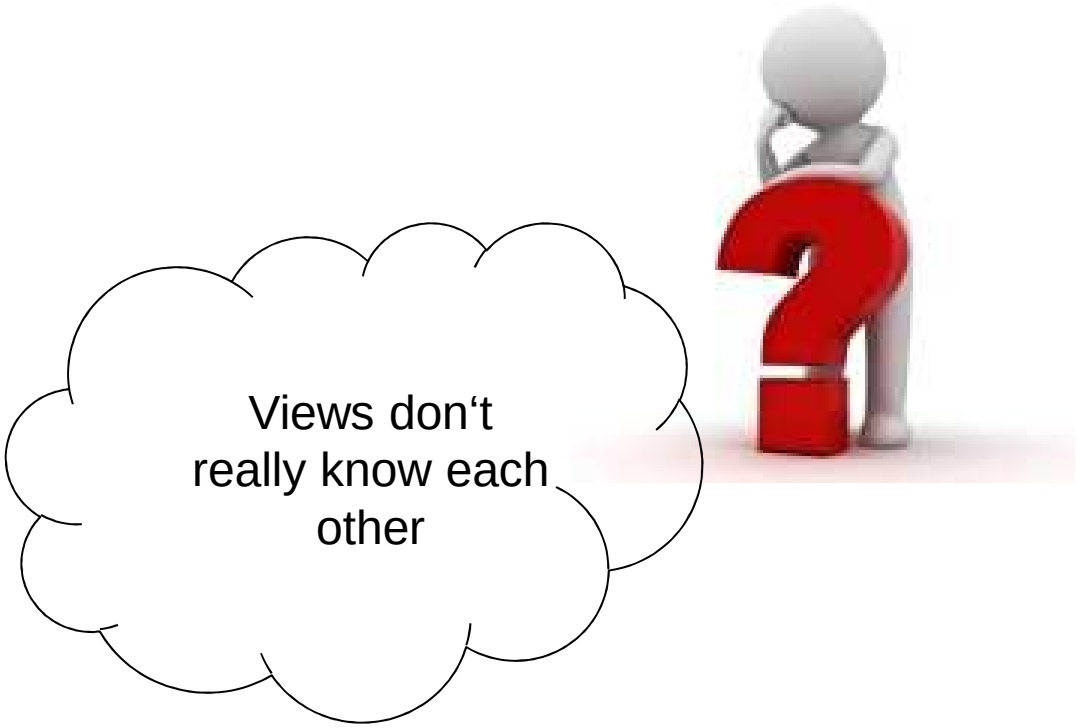
BlueBerry SelectionService

SelectionProvider & SelectionListener

dkfz.
GERMAN
CANCER RESEARCH CENTER
IN THE HELMHOLTZ ASSOCIATION

Motivation

How does the communication work between two Views?



A View is… An Editor is…

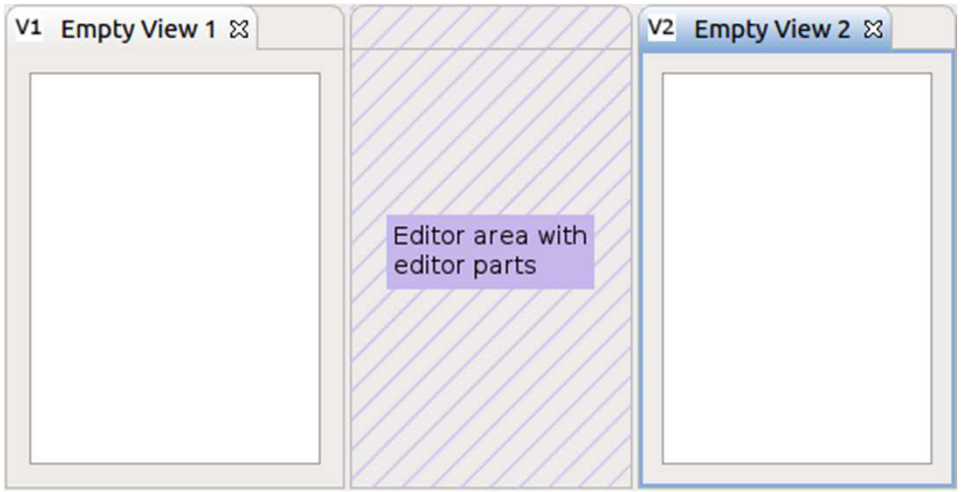
A View is a class which…

- is a subclass of QmitkAbstractView (berry::QtViewPart) &
- has an unique VIEW_ID

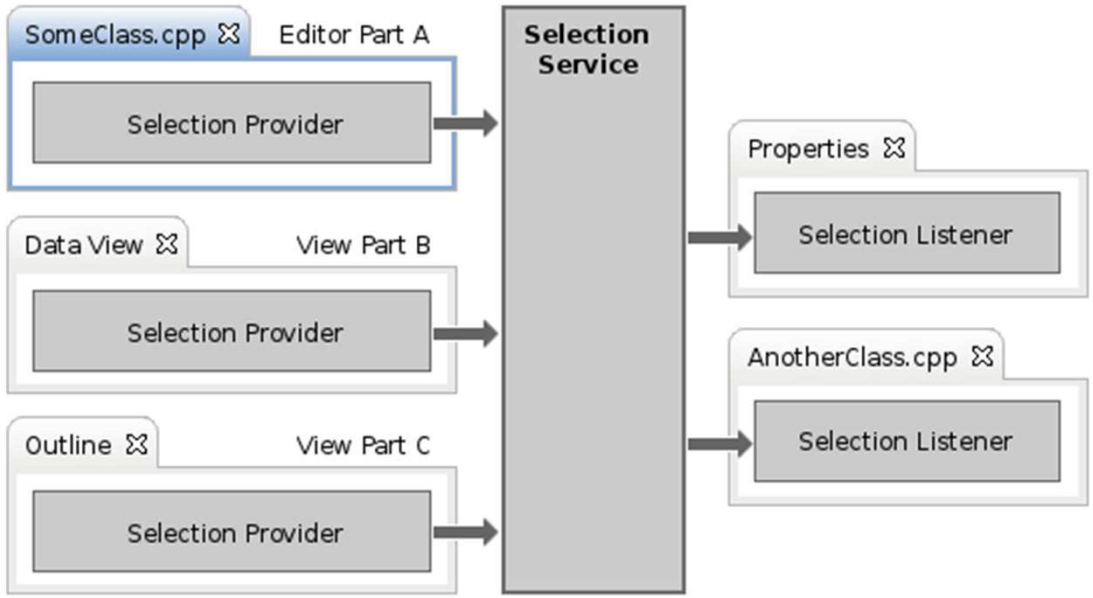
An Editor is a class which…

- is a subclass of berry::QtEditorPart &
- has an unique EDITOR_ID

They are both called Part.



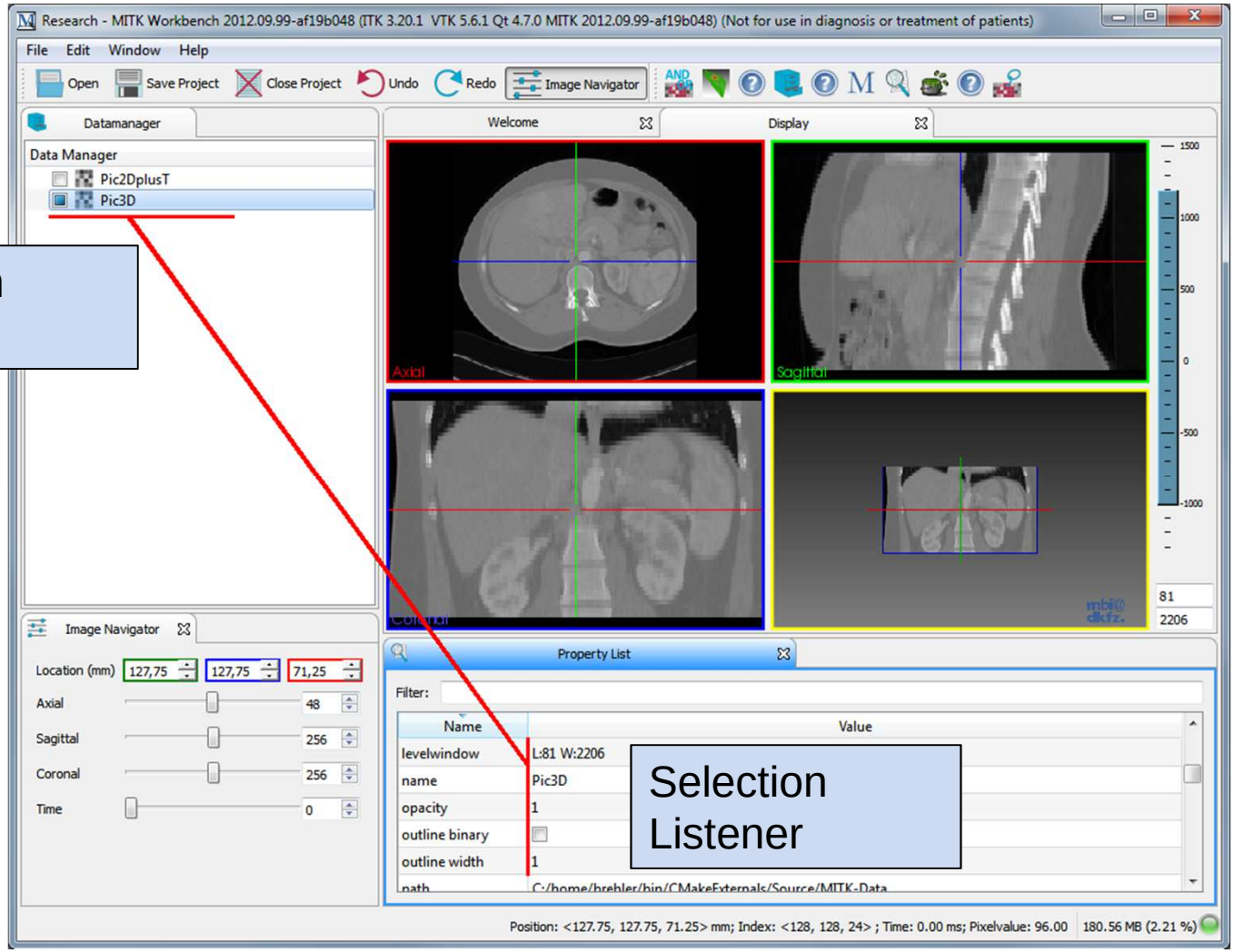
Selection Service in general



Selection Service Diagram

The Selection Service tracks the selection of the currently active part and propagates selection changes to all registered listeners.

Selection Service in MITK



What can be selected?

A selection is a set of highlighted entries.

There are two fundamental different kinds of selections:

- List of objects
- Piece of text (not available in MITK)

Implementation – Qt Selection Provider

Make your View to a Qt Selection Provider
For list- or treeViews

Changes in the HEADER file:

```
berry::QtSelectionProvider::Pointer m_SelectionProvider;
```

Changes in the CPP file (CreateQtPartControl method) :

```
// create new qt selection provider
m_SelectionProvider = new berry::QtSelectionProvider();
// set the item selection model to the model of the QListWidget
m_SelectionProvider->SetItemSelectionModel(m_Controls.m_SelectionList->selectionModel());
// register selection provider
GetSite()->SetSelectionProvider(m_SelectionProvider);
```

The DataNode Selection Provider is realised in the QmitkAbstractView.

Implementation – Qt Selection Listener

Make your View to a Qt Selection Listener

For list- or treeViews

Changes in the HEADER file:

```
void SelectionChanged(berry::IWorkbenchPart::Pointer sourcepart,
                      berry::ISelection::ConstPointer selection);

berry::ISelectionListener::Pointer m_SelectionListener;
```

Changes in the CPP file:

```
void ListenerView::SelectionChanged(berry::IWorkbenchPart::Pointer sourcepart,
                                    berry::ISelection::ConstPointer selection)
{
  // check for null selection
  if (selection.IsNull())
  {
    return;
  }
  // exclude own selection events and check whether this kind of selection can be handled
  if (sourcepart != this &&
      selection.Cast<const berry::IStructuredSelection>())
  {
     // Do something
  }
}
```

The DataNode Selection Listener is also realised in the QmitkAbstractView.

Outlook into the future

An alternative method for interaction between two Views would be the ctkEvent Bus.

This is a suggestion for the next Bugsquashing lecture.

You can find an example in the QmitkDicomEditor.

Source:
../MITK/Plugins/org.mitk.gui.qt.dicom/src/internal/

- QmitkDicomDataEventPublisher
- DicomEventHandler
- QmitkDicomDirectoryListener
- QmitkDicomEditor

Important Links (Wiki & Doxygen Documentation)

Dokumentation

- http://www.mitk.org/Article_Using_the_Selection_Service
- http://docs.mitk.org/2013.09/BlueBerryExampleSelectionServiceQt.html
- http://docs.mitk.org/2013.09/BlueBerryExampleSelectionServiceMitk.html

Examples von Michael Brehler

- https://github.com/MITK/MITK/blob/master/Examples/Plugins/org.mitk.example.gui.selectionservicemitk.views/src/internal
- https://github.com/MITK/MITK/blob/master/Examples/Plugins/org.mitk.example.gui.selectionservicegt/src/internal

Are there any questions?