BS Seminar
Howto Undo / Redo

Do not Panic!

dkfz. GERMAN CANCER RESEARCH CENTER IN THE HELMHOLTZ ASSOCIATION

Undo/Redo in MITK

dkfz.

Simple Concept

1. encapsulate actions as operations
2. define inverse operations
3. keep a list of everything that happened
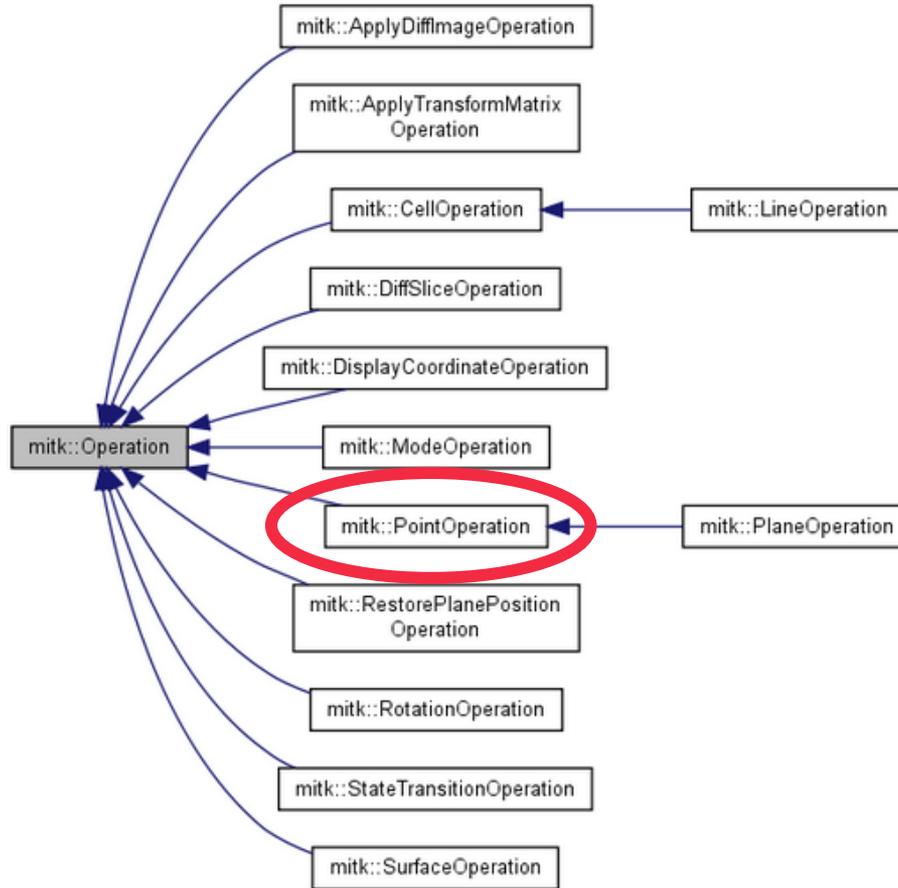
Step 1 – Use Operations

dkfz.

```
mitk::PointSet pointSet;
```

Convetional way

```
mitk::Point3D point;
point1.Fill(5);
pointSet->InsertPoint(position,point1,timestep);
```

Operation – way

```
PointOperation* doOp = new
mitk::PointOperation(OpINSERT,timeInMs, point1,
                     position);
```

```
pointSet->ExecuteOperation(doOp);
```

$$\left(\begin{array}{l} \texttt{PointOperation* doOp = new} \\ \texttt{mitk::}\textbf{PointOperation}\texttt{(OpINSERT,timeInMs, point1,} \\ \qquad\qquad\qquad\textbf{position)}\texttt{;} \end{array} \right)^{-1}$$

$\rightarrow$

```
PointOperation *undoOp = new
mitk::PointOperation(OpREMOVE, timeInMs, point1,
                     position);
```

dkfz.

Bundle both Operations in a OperationEvent:

```
OperationEvent *operationEvent = new
OperationEvent(pointSet, doOp, undoOp, "Add point");
```

Put it into the UndoController:

```
m_UndoController->SetOperationEvent(operationEvent);
```

Undo Controller

- has a static list of operations
- Directly accessible in DataInteractors (and all derived Interactor) as `m_UndoController`

- Commented Step by Step Implementation in:
  mitkPointSetDataInteractor.cpp

  and

  http://docs.mitk.org/nightly-qt4/InteractionPage.html