Proper use of mitk::PointSet

BugSquashing Seminar (25.06.14)

mitk::PointSet

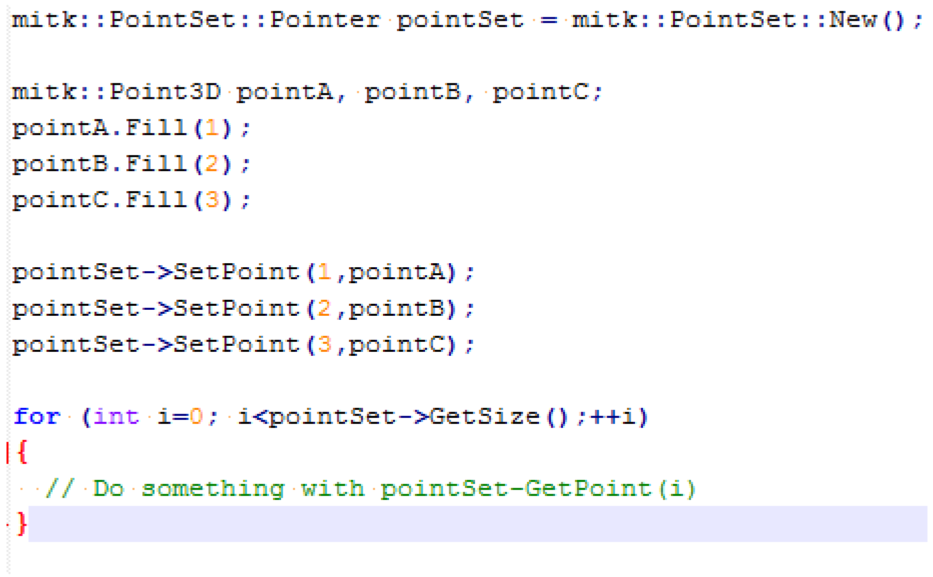
- Keeps and administrates sets of mitk::Point for different timesteps
- std::vector< itk::Mesh<...> > m_PointSetSeries (size equals number of timeSteps)
- One mesh per timestep

PointsContainer

- PointsContainer part of itk::Mesh<…, itk::DefaultDynamicMeshTraits>
- PointsContainerType determined by itk::DefaultDynamicMeshTraits
  - itk::MapContainer
  - (would be an itk::VectorContainer for itk::DefaultStaticMeshTraits, but is not for the mitk::PointSet)

mitk::PointSet is regarded as a map

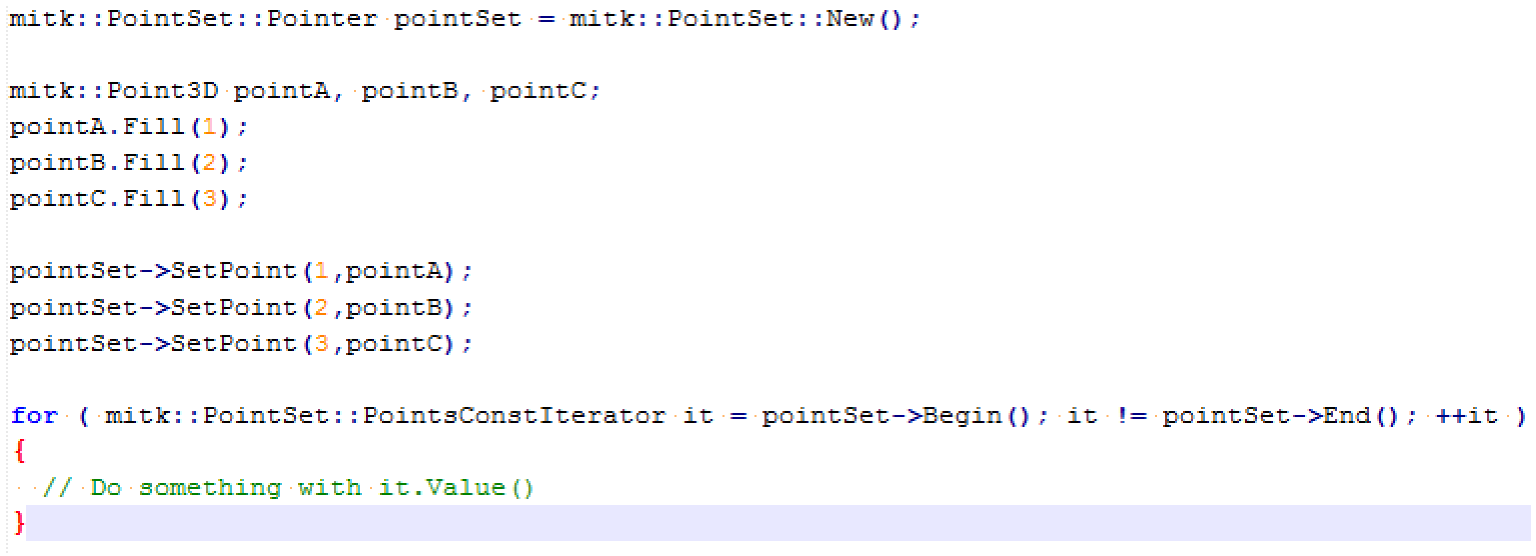
- Cave: mitk::PointSet interfaced like a map, not like a vector
- Common mistake:



```
mitk::PointSet::Pointer pointSet = mitk::PointSet::New();

mitk::Point3D pointA, pointB, pointC;
pointA.Fill(1);
pointB.Fill(2);
pointC.Fill(3);

pointSet->SetPoint(1,pointA);
pointSet->SetPoint(2,pointB);
pointSet->SetPoint(3,pointC);

for (int i=0; i<pointSet->GetSize();++i)
{
  // Do something with pointSet-GetPoint(i)
}
```

mitk::PointSet: use iterators

- Use iterators instead:

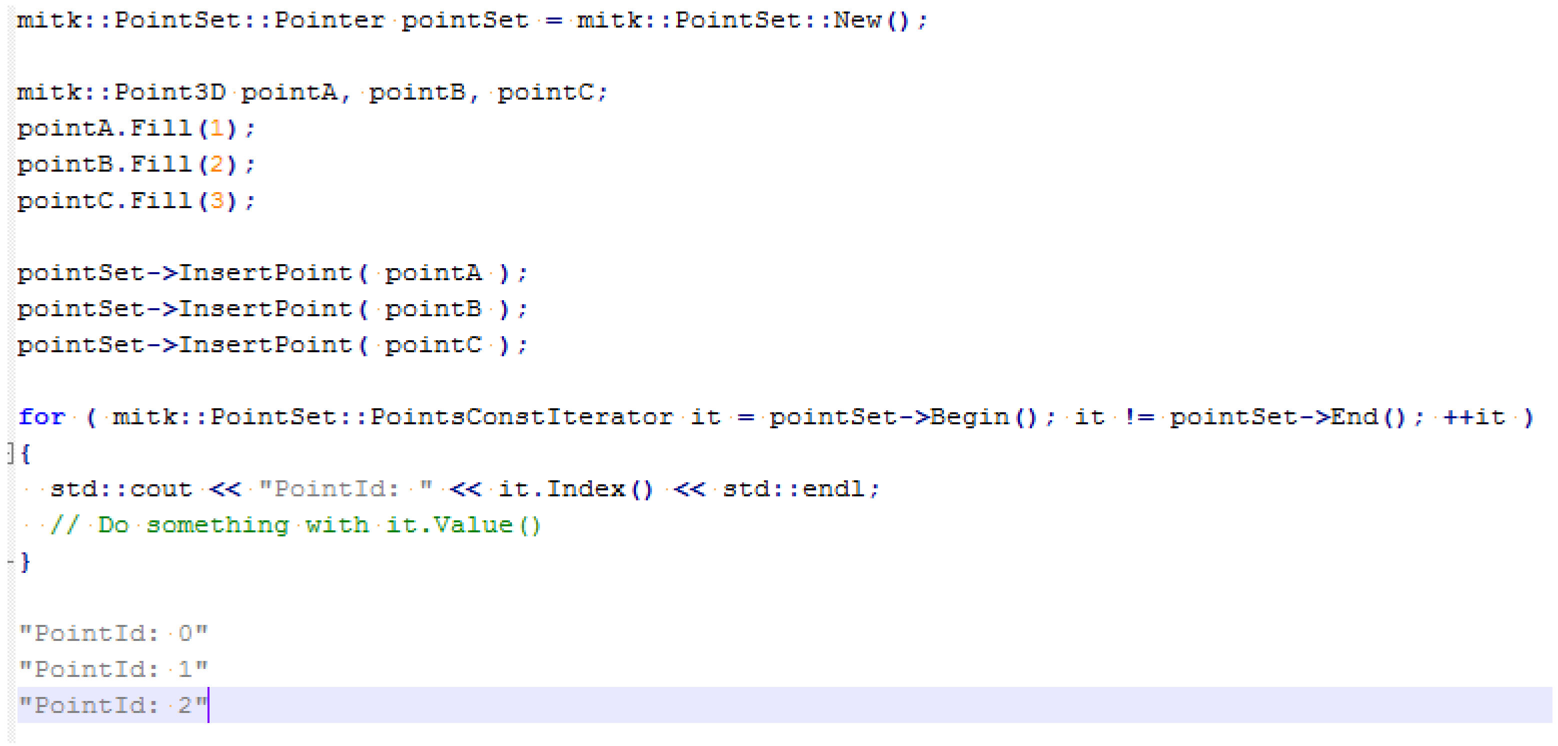


```
mitk::PointSet::Pointer pointSet = mitk::PointSet::New();

mitk::Point3D pointA, pointB, pointC;
pointA.Fill(1);
pointB.Fill(2);
pointC.Fill(3);

pointSet->SetPoint(1,pointA);
pointSet->SetPoint(2,pointB);
pointSet->SetPoint(3,pointC);

for ( mitk::PointSet::PointsConstIterator it = pointSet->Begin(); it != pointSet->End(); ++it )
{
  // Do something with it.Value()
}
```

- Default timestep equals 0
  - Call Begin(int timestep) and End(int timestep) for an arbitrary timestep
  - Returned iterators equal pointSet->End(t0) for any empty timestep t0

mitk::PointSet: sequential point insertion

- Use in a quasi-vector like manner:

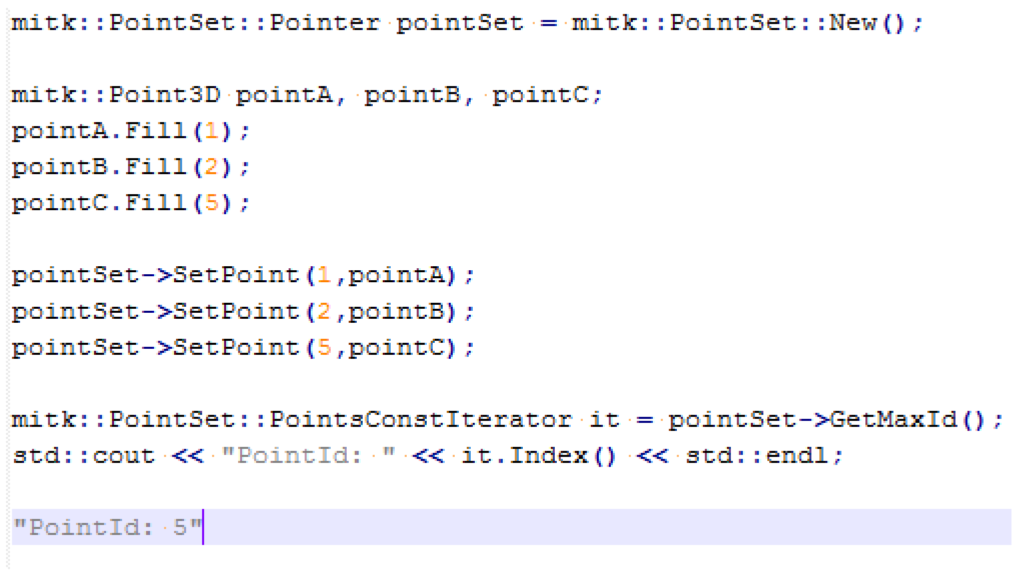


```
mitk::PointSet::Pointer pointSet = mitk::PointSet::New();

mitk::Point3D pointA, pointB, pointC;
pointA.Fill(1);
pointB.Fill(2);
pointC.Fill(3);

pointSet->InsertPoint( pointA );
pointSet->InsertPoint( pointB );
pointSet->InsertPoint( pointC );

for ( mitk::PointSet::PointsConstIterator it = pointSet->Begin(); it != pointSet->End(); ++it )
{
  std::cout << "PointId: " << it.Index() << std::endl;
  // Do something with it.Value()
}

"PointId: 0"
"PointId: 1"
"PointId: 2"
```

- Call InsertPoint(PointType pt, int timestep) for arbitrary timesteps

mitk::PointSet: maxId element

- Get iterator to maxId point:



```
mitk::PointSet::Pointer pointSet = mitk::PointSet::New();

mitk::Point3D pointA, pointB, pointC;
pointA.Fill(1);
pointB.Fill(2);
pointC.Fill(5);

pointSet->SetPoint(1,pointA);
pointSet->SetPoint(2,pointB);
pointSet->SetPoint(5,pointC);

mitk::PointSet::PointsConstIterator it = pointSet->GetMaxId();
std::cout << "PointId: " << it.Index() << std::endl;

"PointId: 5"
```

- GetMaxId(int timestep) for arbitrary timsteps accordingly

Questions?