DICOM-Viewer Technologie & Architektur

Übersicht und VTK.js

Yanic Bernhard

30. April 2025

Ablauf

01

Architektur und verwendete Technologien

02

VTK.js

03

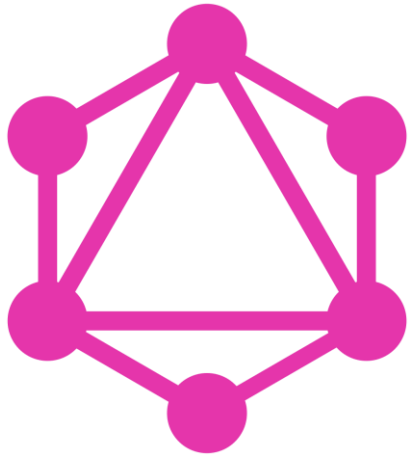
PaintingLayer

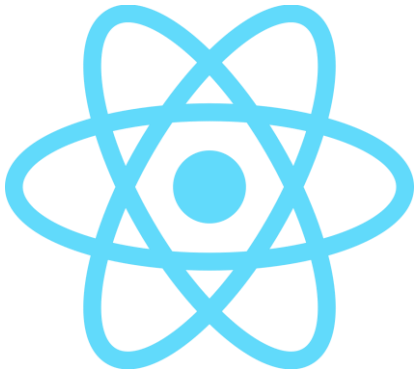
04

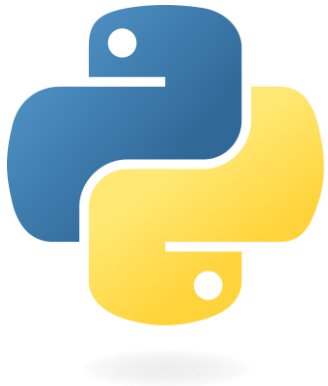
Zusammen-fassung & Fragen

Architektur und verwendete Technologien

- Stack vorgegeben
- Vorteile
- Nachteile

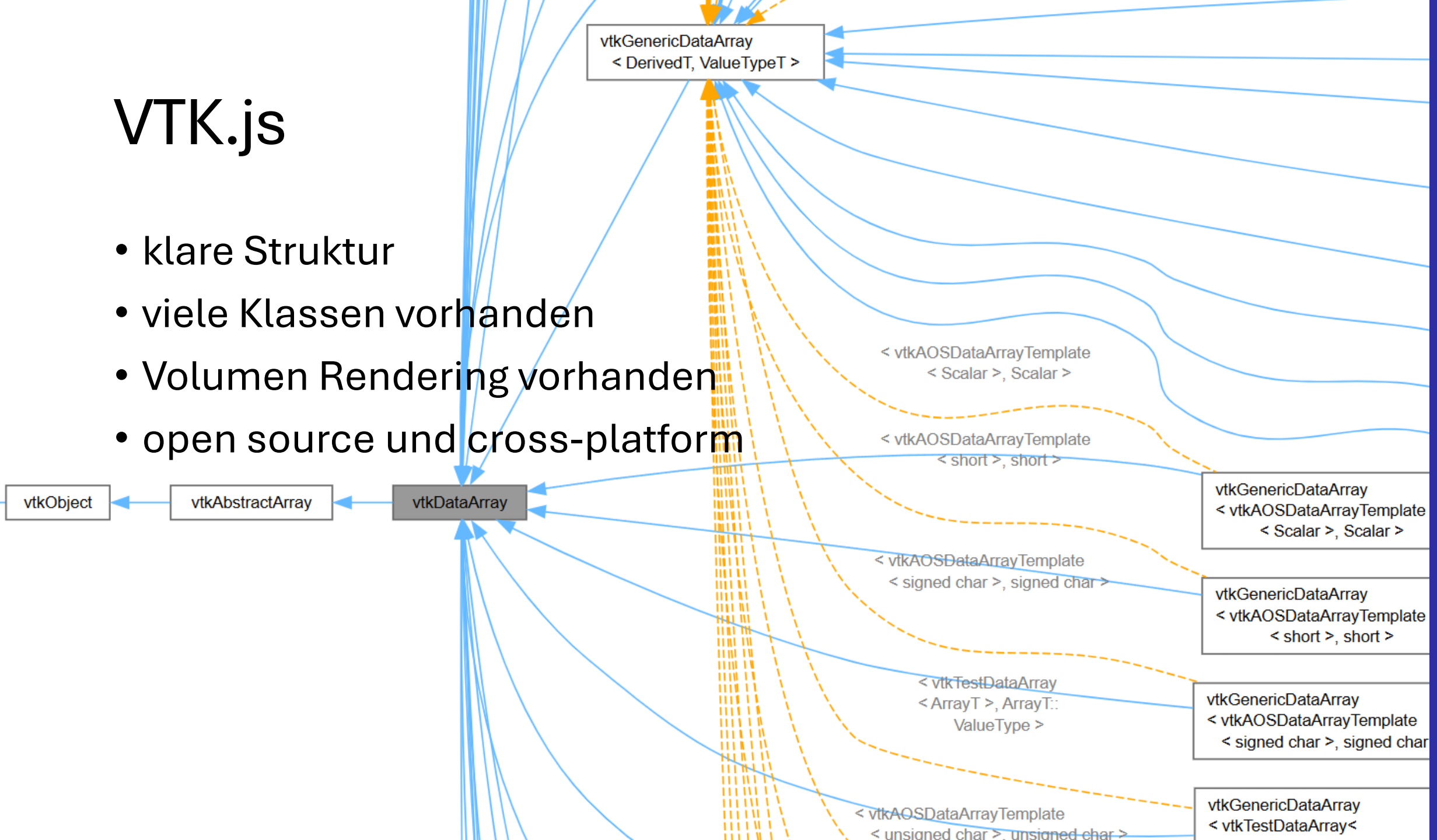




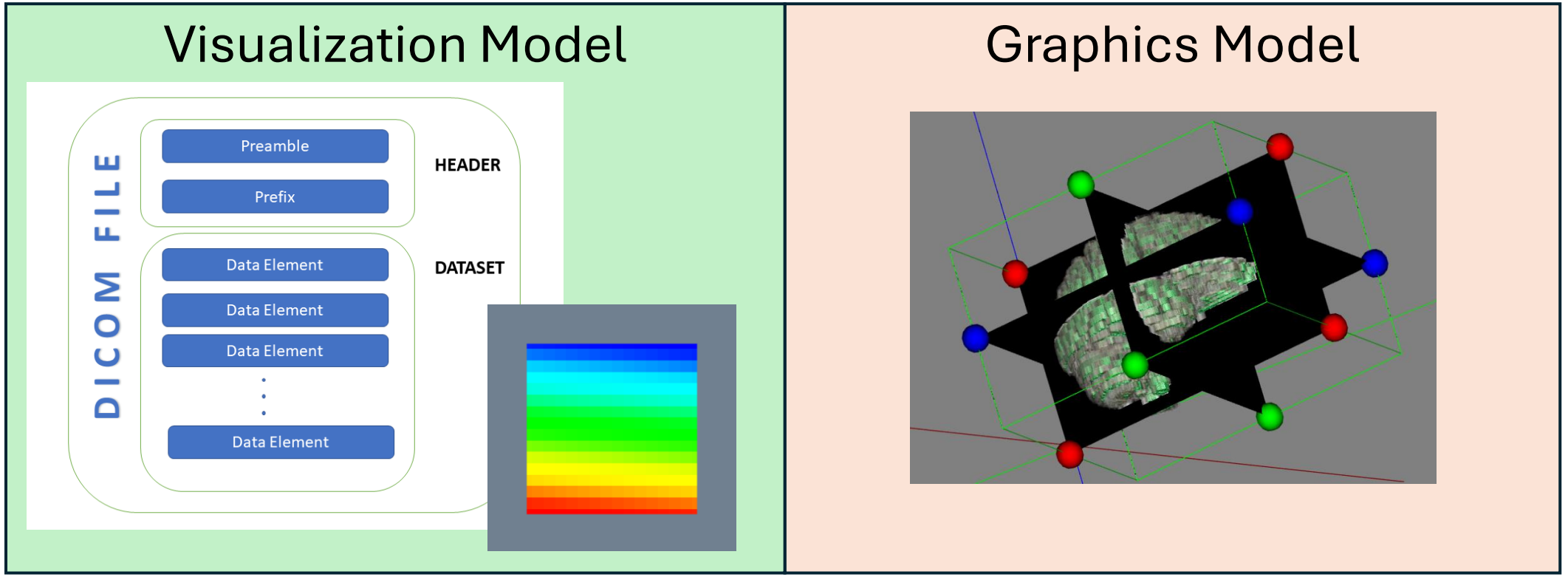


VTK.js

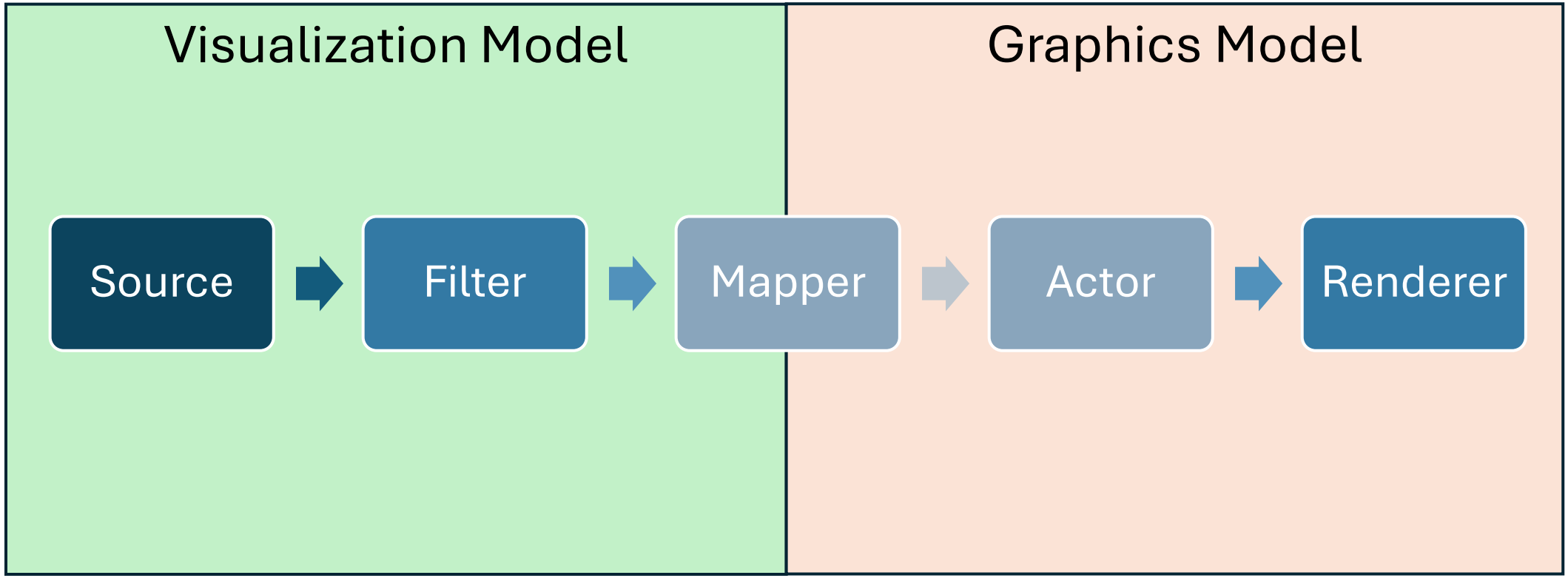
- klare Struktur
- viele Klassen vorhanden
- Volumen Rendering vorhanden
- open source und cross-platform



- Visualization Toolkit

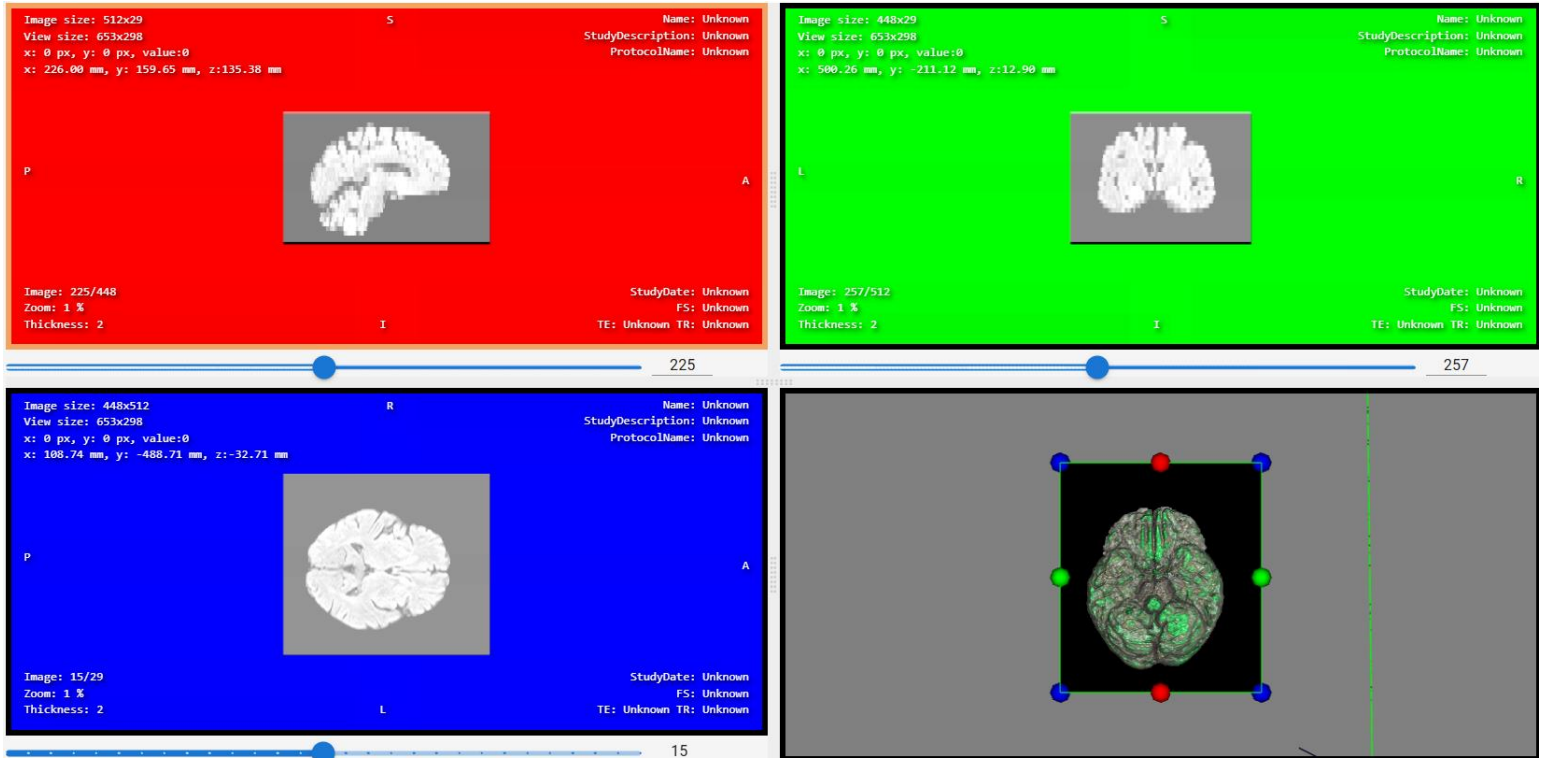


- Visualization Pipeline



PaintingLayer.js

- Performance
- Trennung von Segmentierungs- und Bilddaten



- bestehende Daten importieren oder Daten erstellen.
- [0,0,0,0,0,0,0,0,0]

```
// Create a blank array filled with zeros
const numPaintablePixels = this.imSize.width * this.imSize.height;
this.paintingData = new Uint32Array(numPaintablePixels).fill(0);

// Create VTK data array
this.dataArray = vtkDataArray.newInstance({
  name: 'paintingData',
  values: this.paintingData,
  numberOfComponents: 1
});
```

- in: Datenobjekt
- out: Datenobjekt
- mehrere Filter sind möglich

```
this.reslicer = vtkImageReslice.newInstance(viewMode);
this.reslicer.setSlabMode(SlabMode.MAX);
this.reslicer.setSlabNumberOfSlices(1);
this.reslicer.setTransformInputSampling(false);
this.reslicer.setAutoCropOutput(true);
this.reslicer.setOutputDimensionality(2);
this.reslicer.setInputData(this.imageData);

// Create color mapping pipeline using vtkImageMapToColors
this.imageMapToColors = vtkImageMapToColors.newInstance();
this.imageMapToColors.setInputConnection(this.reslicer.getOutputPort());
```

- Verbindungsstück von visualization and graphics Modell
- Wird mit dem letzten Filter verbunden

```
// Create mapper for the color-mapped output
this.mapper = vtkImageMapper.newInstance();
this.mapper.setInputConnection(this.imageMapToColors.getOutputPort());
```

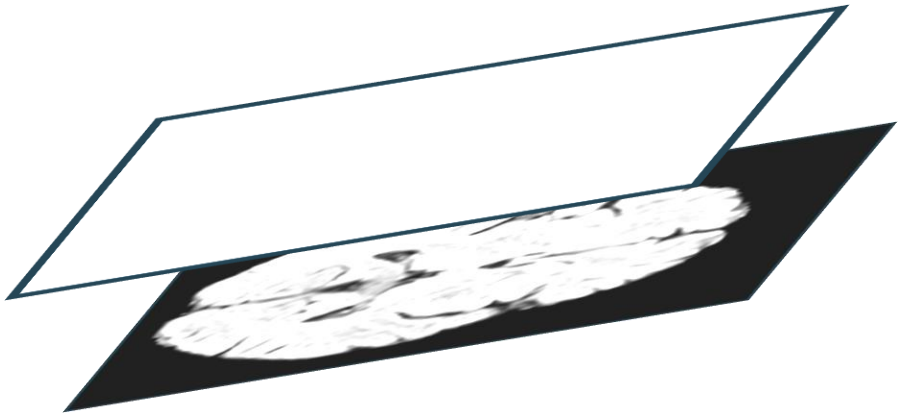
```
// Create actor for the painting layer
this.actor = vtkImageSlice.newInstance();
this.actor.setMapper(this.mapper);
```

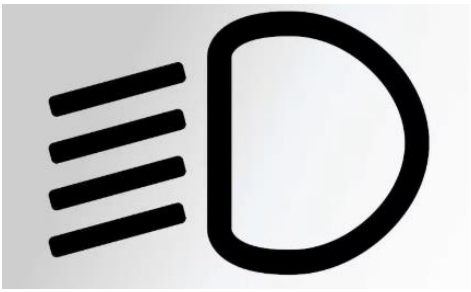
- Eigenschaften
- Geometrie
- Transformationen

```
const transformedPlaneAxes = this.transformPlaneAxes(this.scene.planeAxes, flip.x, flip.y, this.spacing);
// Set the resulting matrix as the user matrix
this.actor.setUserMatrix(transformedPlaneAxes);

this.actor.setScale(this.spacing.x, this.spacing.y, this.spacing.z); // Set scale based on spacing

// Configure properties for the painting actor
const property = this.actor.getProperty();
property.setInterpolationType(InterpolationType.NEAREST);
```

```
// Add actor to renderer - add last to ensure it's on top
this.scene.renderer.addActor(this.actor);
this.actor.setVisibility(true);
```

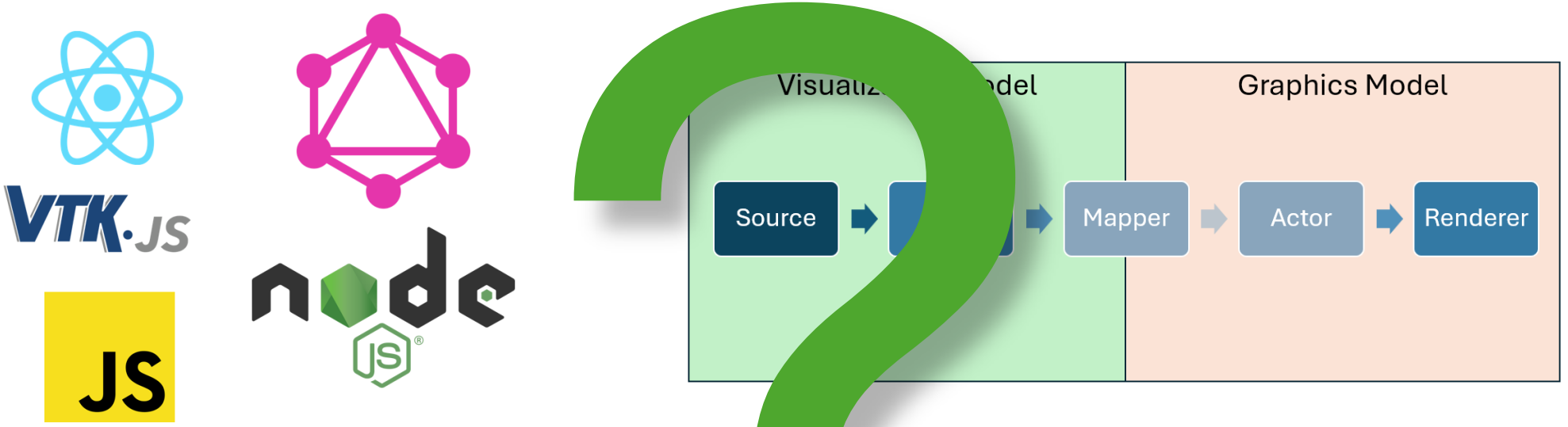
- Actor wird mit den anderen Actors zur Szene hinzugefügt.





- vtkRenderWindow
- Rerendering

Zusammenfassung & Fragen



Tag	Name	Value
(0008, 1030)	Study Description	'CT_ABDOMEN_W_IV_CONTRAST'
(0008, 103e)	Series Description	'ABD'
(0010, 0010)	Patient's Name	'SIMPSON_HOMER_J'
(0010, 0020)	Patient ID	'5553226'
(0020, 000d)	Study Instance UID	1.2.826.0.1.3680043.2.1125.1.38381854871216336385 8062044218957
(0020, 000e)	Series Instance UID	1.2.826.0.1.3680043.2.1125.1.68878959984837726447 6707551399667
(0020, 0013)	Instance Number	"20"
(7fe0, 0010)	Pixel Data	Array of 524288 elements