11/09/2013

Class API Documentation

Alfred Franz

Overview

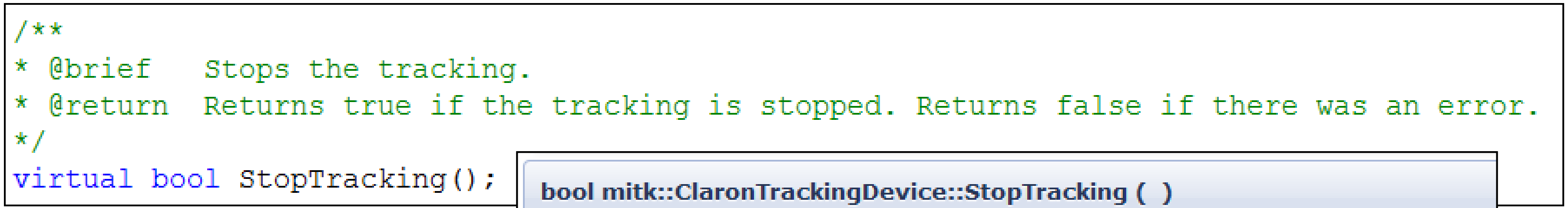
Types of Documentation:

- User Documentation (User Manual)
- Developer Documentation:
  - Meta Documentation (Developers Manual)
  - API Documentation (Documentation of Interfaces/Classes)
  - Code Documentation

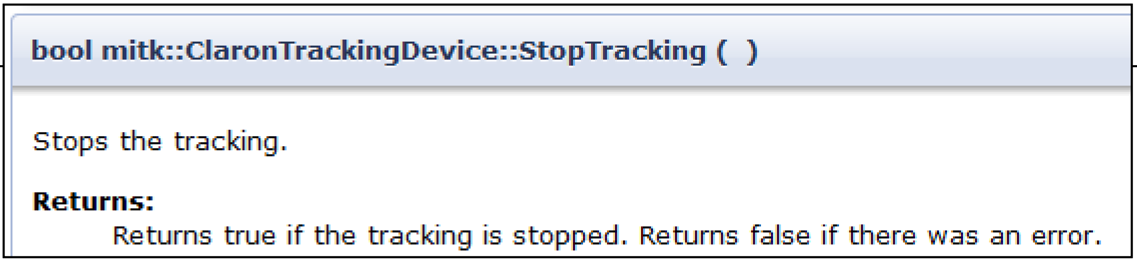
- API Documentation is the most standardized and very important
- it‘s the key for the big goal: **reusability**

Tools

- Well known: Doxygen, Javadoc
- Syntax quite similar
- MITK uses Doxygen
- Example doxygen/javadoc method documentation:



```
/**
* @brief   Stops the tracking.
* @return  Returns true if the tracking is stopped. Returns false if there was an error.
*/
virtual bool StopTracking();
```



bool mitk::ClaronTrackingDevice::StopTracking ( )

Stops the tracking.

Returns:
Returns true if the tracking is stopped. Returns false if there was an error.

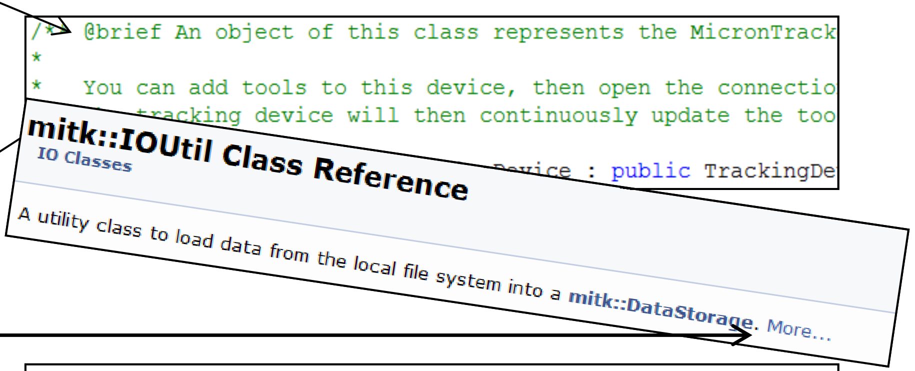
Outline API Documentation

dkfz.

- Description of class

A short descrition, e.g. ‚one sentence („brief“)

More detailed Description.

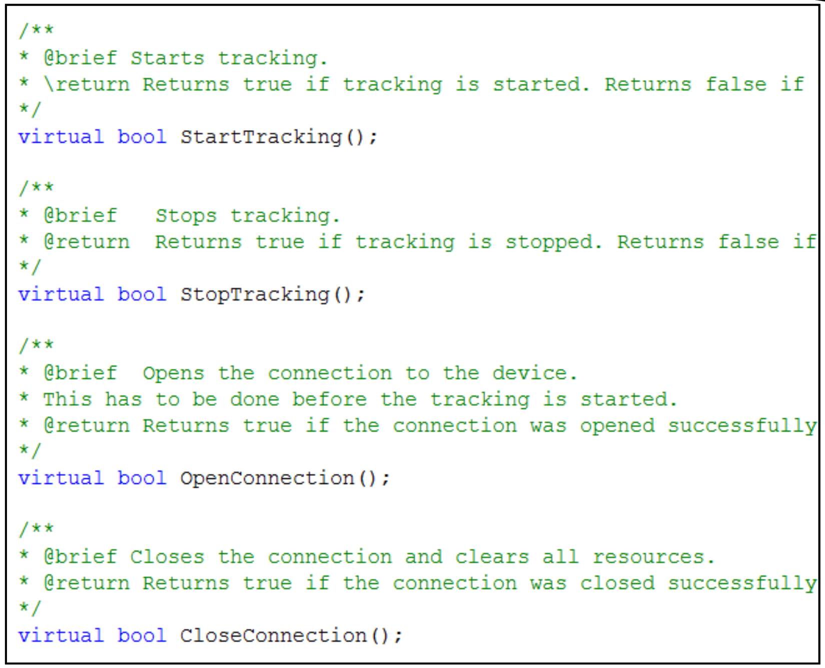


```
/** @brief An object of this class represents the MicronTrack
 *
 *   You can add tools to this device, then open the connectio
 *   tracking device will then continuously update the too
                                 Device : public TrackingDe
```

mitk::IOUtil Class Reference
IO Classes

A utility class to load data from the local file system into a **mitk::DataStorage**. More...

- Description of every public
  - Constructor
  - Method
  - Typedef
  - Variable



```
/**
* @brief Starts tracking.
* \return Returns true if tracking is started. Returns false if
*/
virtual bool StartTracking();

/**
* @brief   Stops tracking.
* @return  Returns true if tracking is stopped. Returns false if
*/
virtual bool StopTracking();

/**
* @brief  Opens the connection to the device.
* This has to be done before the tracking is started.
* @return Returns true if the connection was opened successfully
*/
virtual bool OpenConnection();

/**
* @brief Closes the connection and clears all resources.
* @return Returns true if the connection was closed successfully
*/
virtual bool CloseConnection();
```

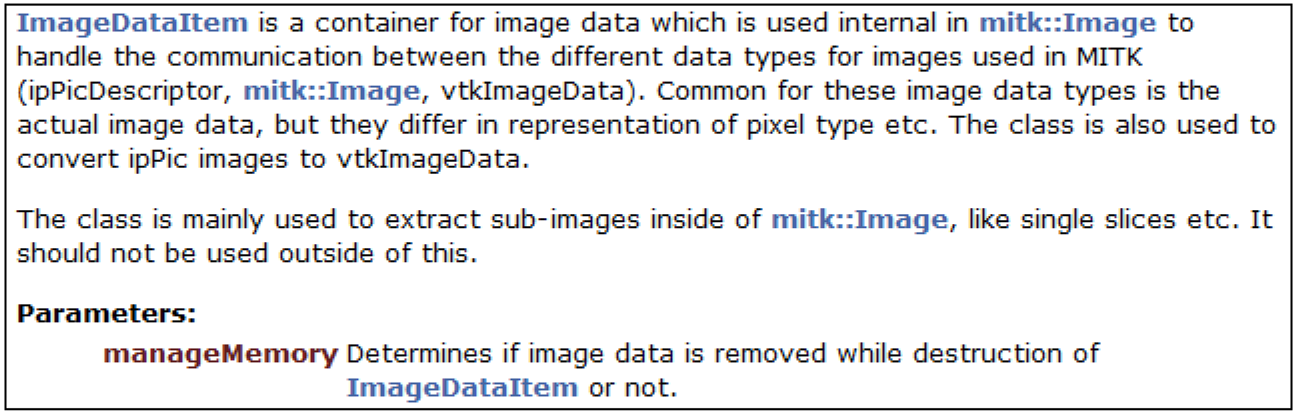
Class API Documentation: Class Description

dkfz.

- What represents one object of this class?
- What is the purpose of such an object?
- What preferences does an object have?
- Maybe: a small code example how to use it

→ Very good examples: Qt!

Example from MITK:



ImageDataItem is a container for image data which is used internal in **mitk::Image** to handle the communication between the different data types for images used in MITK (ipPicDescriptor, **mitk::Image**, vtkImageData). Common for these image data types is the actual image data, but they differ in representation of pixel type etc. The class is also used to convert ipPic images to vtkImageData.

The class is mainly used to extract sub-images inside of **mitk::Image**, like single slices etc. It should not be used outside of this.

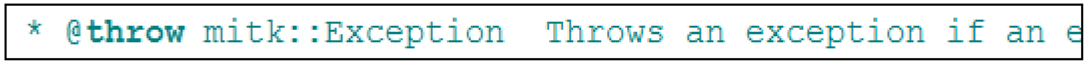
Parameters:

manageMemory	Determines if image data is removed while destruction of **ImageDataItem** or not.

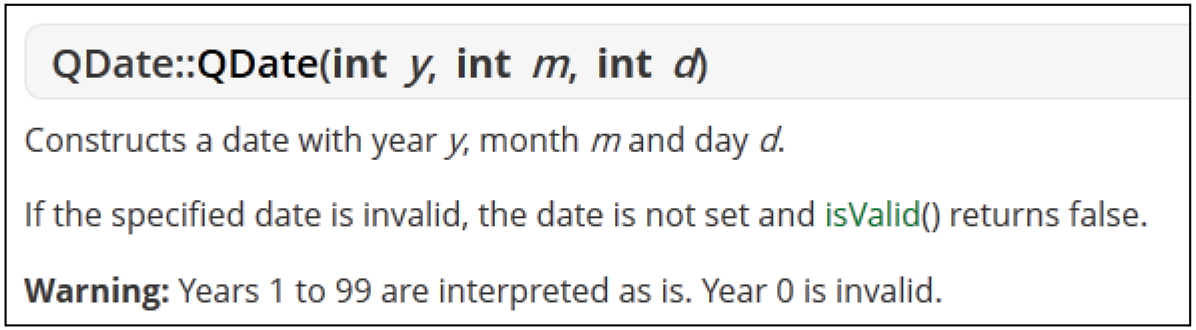
Class API Documentation: Constructor

Constructor:

- state of an object after instantiation
- more initialization nessecary?
- documentation of parameters
- don‘t forget possible exceptions



```
* @throw mitk::Exception  Throws an exception if an e
```



QDate::QDate(int *y*, int *m*, int *d*)

Constructs a date with year *y*, month *m* and day *d*.

If the specified date is invalid, the date is not set and isValid() returns false.

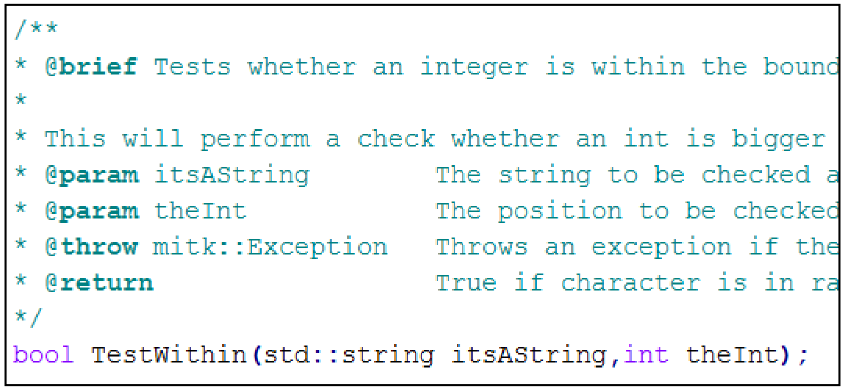
Warning: Years 1 to 99 are interpreted as is. Year 0 is invalid.

Class API Documentation: Methods

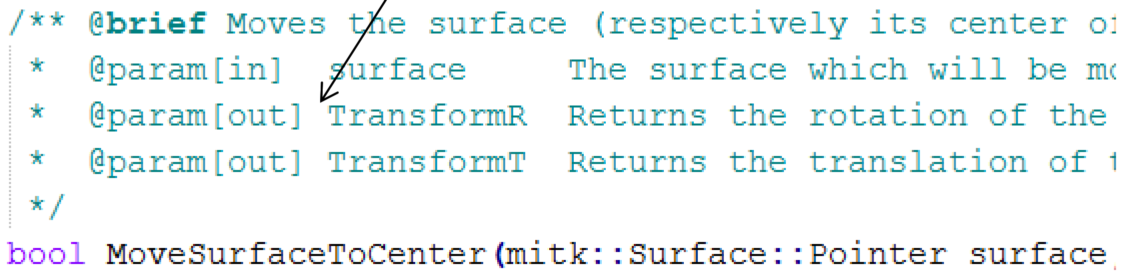
Methods:

- What is the effect of this method?
- What happens if there is an error?
- What is the return value?
- Describe all parameters
  - Are they return values?

→ You should use the @param[out] syntax to mark them



```
/**
* @brief Tests whether an integer is within the bound
*
* This will perform a check whether an int is bigger
* @param itsAString        The string to be checked a
* @param theInt            The position to be checked
* @throw mitk::Exception   Throws an exception if the
* @return                  True if character is in ra
*/
bool TestWithin(std::string itsAString,int theInt);
```



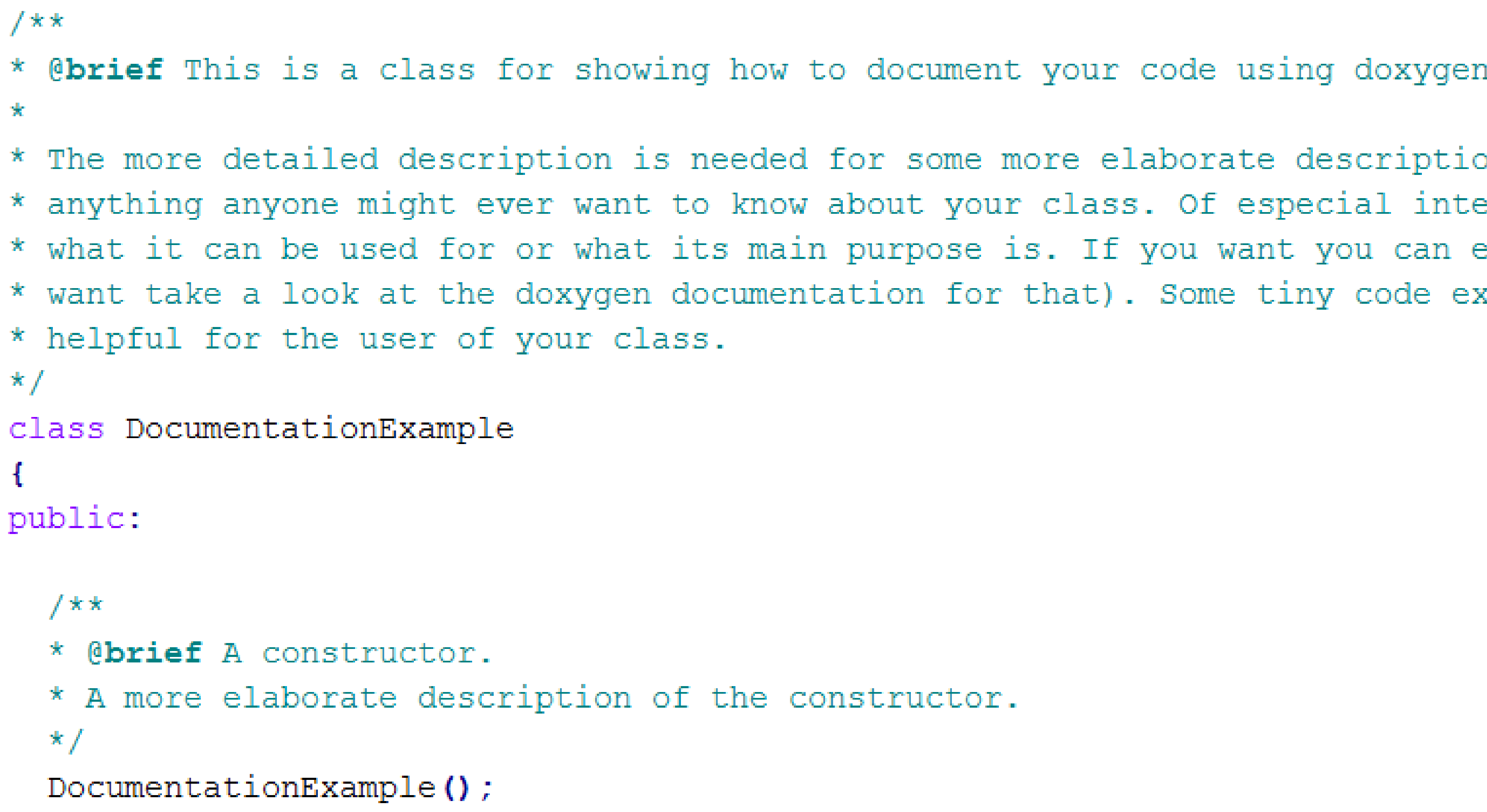
```
/** @brief Moves the surface (respectively its center o
 *  @param[in]  surface     The surface which will be mo
 *  @param[out] TransformR  Returns the rotation of the
 *  @param[out] TransformT  Returns the translation of t
 */
bool MoveSurfaceToCenter(mitk::Surface::Pointer surface
```

Alfred Michael Franz
MBI

Class API Documentation: Example File

dkfz.

MITK/Examples/DocumentationExample.h



```
/**
* @brief This is a class for showing how to document your code using doxygen
*
* The more detailed description is needed for some more elaborate descriptio
* anything anyone might ever want to know about your class. Of especial inte
* what it can be used for or what its main purpose is. If you want you can e
* want take a look at the doxygen documentation for that). Some tiny code ex
* helpful for the user of your class.
*/
class DocumentationExample
{
public:

  /**
  * @brief A constructor.
  * A more elaborate description of the constructor.
  */
  DocumentationExample();
```

Thank you for your attention!

Any Questions?

References

- Prof. Heuzeroth - Vorlesungsfolien „Grundlagen der praktischen Informatik 2“, Medizinische Informatik Heidelberg / Heilbronn