Smart Document Engine Library Reference

version 3.2.0

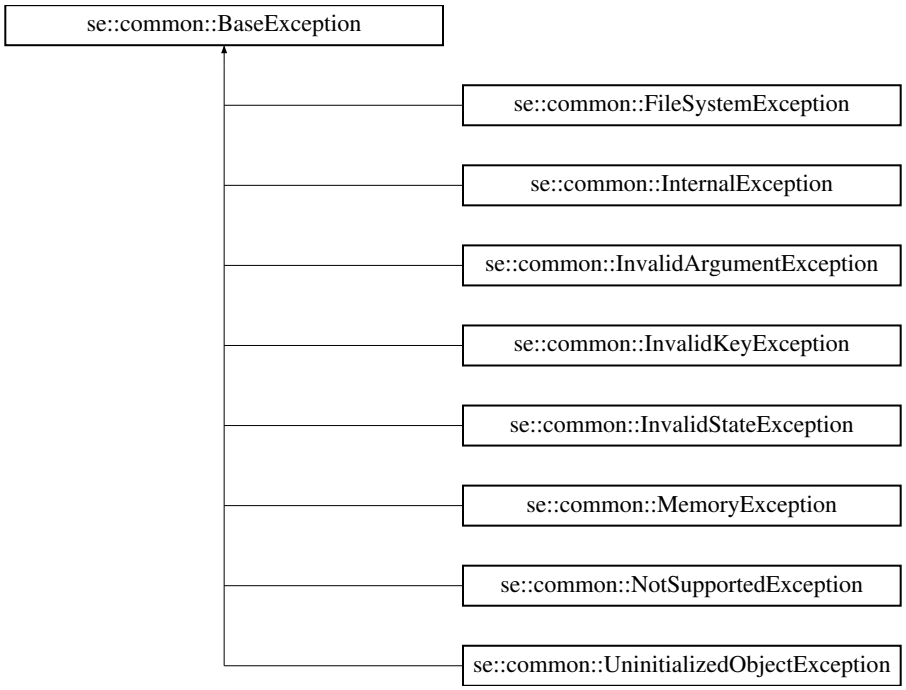
1 Class Documentation

1.1 se::common::BaseException Class Reference

BaseException class - base class for all SE exeptions. Cannot be created directly.

`#include <se_exception.h>`

Inheritance diagram for se::common::BaseException:



Public Member Functions

- virtual ~**BaseException** ()

  Non-trivial dtor.
- **BaseException** (const BaseException &copy)

  Copy ctor.
- virtual const char * ExceptionName () const

  Returns exception class name.
- virtual const char * **what** () const

  Returns exception message.

Protected Member Functions

- **BaseException** (const char *msg)

  Protected ctor.

Private Attributes

- char * msg_
  stored exception message

1.1.1 Detailed Description

BaseException class - base class for all SE exeptions. Cannot be created directly.

Definition at line 22 of file se_exception.h.

1.1.2 Member Function Documentation

ExceptionName()

```
virtual const char * se::common::BaseException::ExceptionName ( ) const  [virtual]
```

Returns exception class name.

Reimplemented in se::common::InvalidKeyException, se::common::NotSupportedException, se::common::FileSystemException, se::common::UninitializedObjectException, se::common::InvalidArgumentException, se::common::MemoryException, se::common::InvalidStateException, and se::common::InternalException.

1.1.3 Member Data Documentation

msg_

```
char* se::common::BaseException::msg_  [private]
```

stored exception message

Definition at line 41 of file se_exception.h.

1.2 se::common::ByteString Class Reference

Class representing byte string.

```
#include <se_string.h>
```

Public Member Functions

- **ByteString** ()
  Default ctor, creates an empty string.
- ~**ByteString** ()
  Non-trivial dtor.
- **ByteString** (const unsigned char *bytes, size_t n)
  Ctor from a given sequence of bytes and length.
- **ByteString** (const ByteString &other)
  Copy ctor.
- ByteString & **operator=** (const ByteString &other)
  Assignment operator.
- void **swap** (ByteString &other) noexcept
  Swap.
- int **GetLength** () const noexcept
  Returns the number of bytes.
- int **GetRequiredBase64BufferLength** () const
  Returns length of base64 formated buffer.
- int **CopyBase64ToBuffer** (char *out_buffer, int buffer_length) const
  Format buffer to base64.
- MutableString **GetBase64String** () const
  Get base64 string from buffer.
- int **GetRequiredHexBufferLength** () const
  Returns length of hex formated buffer.
- int **CopyHexToBuffer** (char *out_buffer, int buffer_length) const
  Format buffer to hex.
- MutableString **GetHexString** () const
  Get hex string from buffer.

Private Attributes

- size_t len_
  length of the internal buffer in bytes
- uint8_t * buf_
  internal buffer

1.2.1 Detailed Description

Class representing byte string.

Definition at line 397 of file se_string.h.

1.2.2 Member Data Documentation

len_

```
size_t se::common::ByteString::len_  [private]
```

length of the internal buffer in bytes

Definition at line 439 of file se_string.h.

buf_

```
uint8_t* se::common::ByteString::buf_  [private]
```

internal buffer

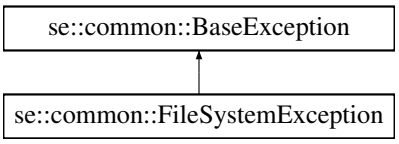
Definition at line 440 of file se_string.h.

1.3 se::common::FileSystemException Class Reference

FileSystemException: thrown if an attempt is made to read from a non-existent file, or other file-system related IO error.

```
#include <se_exception.h>
```

Inheritance diagram for se::common::FileSystemException:



Public Member Functions

- **FileSystemException** (const char *msg)
  Ctor with an exception message.
- **FileSystemException** (const FileSystemException &copy)
  Copy ctor.
- virtual ~**FileSystemException** () override=default
  Default dtor.
- virtual const char * ExceptionName () const override
  Returns exception class name.

Public Member Functions inherited from se::common::BaseException

- virtual ~**BaseException** ()
  Non-trivial dtor.
- **BaseException** (const BaseException &copy)
  Copy ctor.
- virtual const char * **what** () const
  Returns exception message.

Additional Inherited Members

Protected Member Functions inherited from se::common::BaseException

- **BaseException** (const char *msg)
  Protected ctor.

1.3.1 Detailed Description

FileSystemException: thrown if an attempt is made to read from a non-existent file, or other file-system related IO error.

Definition at line 92 of file se_exception.h.

1.3.2 Member Function Documentation

ExceptionName()

```
virtual const char * se::common::FileSystemException::ExceptionName ( ) const  [override],
[virtual]
```

Returns exception class name.

Reimplemented from se::common::BaseException.

1.4 se::common::Image Class Reference

Class representing bitmap image.

```
#include <se_image.h>
```

Public Member Functions

- virtual ~**Image** ()=default
  Default dtor.
- virtual int GetNumberOfLayers () const =0
  Gets the number of additional layers.
- virtual const Image & GetLayer (const char *name) const =0
  Gets the additional layer by the specified name.
- virtual const Image * GetLayerPtr (const char *name) const =0
  Gets the additional layer by the specified name.
- virtual ImagesMapIterator LayersBegin () const =0
  Gets the 'begin' map iterator to the internal layers collection.
- virtual ImagesMapIterator LayersEnd () const =0
  Gets the 'end' map iterator to the internal layers collection.
- virtual bool HasLayer (const char *name) const =0
  Checks whether the Image contains the layer with the specified name.
- virtual bool HasLayers () const =0
  Checks whether the Image contains the layers.
- virtual void RemoveLayer (const char *name)=0
  Removes the layer with the specified name.
- virtual void **RemoveLayers** ()=0
  Clears the internal layers collection.
- virtual void SetLayer (const char *name, const Image &image)=0
  Add the image with the specified name to the internal layers collection with copying of the pixels of the given image.
- virtual void SetLayerWithOwnership (const char *name, Image *image)=0

Add the image with the specified name to the internal layers collection by transfering the given image to the internal layers collection. The caller has to release the ownership of the set image.

- virtual Image * CloneDeep () const =0

  Clones an image with copying of all pixels.
- virtual Image * CloneShallow () const =0

  Clones an image without copying the pixels. The cloned image will be a separate object without memory ownership, the operations with it will be invalid if the source is deallocated.
- virtual void **Clear** ()=0

  Clears the internal image structure.
- virtual int GetRequiredBufferLength () const =0

  Gets the required buffer length for copying the image pixels into an external pixels buffer.
- virtual int CopyToBuffer (unsigned char *buffer, int buffer_length) const =0

  Copies the image pixels.
- virtual void Save (const char *image_filename) const =0

  Saves the image to an external file (png, jpg, tif). Format is deduced from the filename extension.
- virtual int GetRequiredBase64BufferLength () const =0

  Returns required buffer size for Base64 JPEG representation of an image. WARNING: will perform one extra JPEG encoding of an image.
- virtual int CopyBase64ToBuffer (char *out_buffer, int buffer_length) const =0

  Copies the Base64 JPEG representation of an image to an external buffer.
- virtual MutableString GetBase64String () const =0

  Returns Base64 JPEG representation of an image.
- virtual double EstimateFocusScore (double quantile=0.95) const =0

  Estimates focus score of an image.
- virtual void Resize (const Size &new_size)=0

  Scale the image to a new size.
- virtual Image * CloneResized (const Size &new_size) const =0

  Clones the image scaled to a new size.
- virtual void Crop (const Quadrangle &quad)=0

  Projectively crops a region of image, with approximate selection of the cropped image size.
- virtual Image * CloneCropped (const Quadrangle &quad) const =0

  Clones the image projectively cropped with approximate selection of the target image size.
- virtual void Crop (const Quadrangle &quad, const Size &size)=0

  Projectively crops a region of image, with a given target size.
- virtual Image * CloneCropped (const Quadrangle &quad, const Size &size) const =0

  Clones the image projectively cropped with a given target size.
- virtual void Crop (const Rectangle &rect)=0

  Crops an image to a rectangular image region.
- virtual Image * CloneCropped (const Rectangle &rect) const =0

  Clones the image cropped to a selected rectangular region (with copying of pixels)
- virtual Image * CloneCroppedShallow (const Rectangle &rect) const =0

  Clones the image cropped to a selected rectangular region, without copying of pixels. The cloned image will be a separate object without memory ownership, the operations with it will be invalid if the source is deallocated.
- virtual void Mask (const Rectangle &rect, int pixel_expand=0, double pixel_density=0)=0

  Masks image region specified by rectangle.
- virtual Image * CloneMasked (const Rectangle &rect, int pixel_expand=0) const =0

  Clone the image with masked region specified by rectangle.
- virtual void Mask (const Quadrangle &quad, int pixel_expand=0, double pixel_density=0)=0

  Mask image region specified by quadrangle.
- virtual Image * CloneMasked (const Quadrangle &quad, int pixel_expand=0) const =0

  Clone the image with masked region specified by quadrangle.

- virtual void Fill (const Rectangle &rect, int ch1, int ch2=0, int ch3=0, int ch4=0, int pixel_expand=0)=0

  Fills image region specified by rectangle and color. The method will use the first as many channel values as there are channels in the image.
- virtual Image * CloneFilled (const Rectangle &rect, int ch1, int ch2=0, int ch3=0, int ch4=0, int pixel_↩ expand=0) const =0

  Clone the image with filled region specified by rectangle and color. The method will use the first as many channel values as there are channels in the image.
- virtual void Fill (const Quadrangle &quad, int ch1, int ch2=0, int ch3=0, int ch4=0, int pixel_expand=0)=0

  Fill image region specified by quadrangle and color. The method will use the first as many channel values as there are channels in the image.
- virtual Image * CloneFilled (const Quadrangle &quad, int ch1, int ch2=0, int ch3=0, int ch4=0, int pixel_↩ expand=0) const =0

  Clone the image with filled region specified by quadrangle and color. The method will use the first as many channel values as there are channels in the image.
- virtual void **FlipVertical** ()=0

  Flips an image around the vertical axis.
- virtual Image * CloneFlippedVertical () const =0

  Clones the image flipped around the vertical axis.
- virtual void **FlipHorizontal** ()=0

  Flips an image around the horizontal axis.
- virtual Image * CloneFlippedHorizontal () const =0

  Clones the image flipped around the horizontal axis.
- virtual void Rotate90 (int times)=0

  Rotates the image clockwise by a multiple of 90 degrees.
- virtual Image * CloneRotated90 (int times) const =0

  Clones the image rotated clockwise by a multiple of 90 degrees.
- virtual void **AverageChannels** ()=0

  Makes a single-channel image with averaged intensity values.
- virtual Image * CloneAveragedChannels () const =0

  Clones the image with averaged channel intensity values.
- virtual void **Invert** ()=0

  Inverts the colors of the image.
- virtual Image * CloneInverted () const =0

  Clones the image with inverted colos.
- virtual int **GetWidth** () const =0

  Gets the image width in pixels.
- virtual int **GetHeight** () const =0

  Gets the image height in pixels.
- virtual Size **GetSize** () const =0

  Gets the image size in pixels.
- virtual int **GetStride** () const =0

  Gets the number of image row in bytes, including alignment.
- virtual int **GetChannels** () const =0

  Gets the number of channels per pixel.
- virtual void * **GetUnsafeBufferPtr** () const =0

  Gets the pointer to the pixels buffer.
- virtual bool **IsMemoryOwner** () const =0

  Returns whether this instance owns and will release pixel data.
- virtual void **ForceMemoryOwner** ()=0

  Forces memory ownership - allocates new image data and copies the pixels.
- virtual void **Serialize** (Serializer &serializer) const =0

  Serializes the image given the serializer object.

Static Public Member Functions

- static int GetNumberOfPages (const char *image_filename)

  Returns the number of pages in an image.
- static MutableString GetImagePageName (const char *image_filename, int page_number)

  Returns the name of the specified page.
- static Image * CreateEmpty ()

  Factory method for creating an empty image.
- static Image * FromFile (const char *image_filename, const int page_number=0, const Size &max_size=Size(25000, 25000))

  Factory method for loading an image from file. Will be treated as IPF_G or IPF_RGB.
- static Image * FromFileBuffer (unsigned char *data, int data_length, const int page_number=0, const Size &max_size=Size(25000, 25000))

  Factory method for loading an image from file pre-loaded in a buffer Will be treated as IPF_G or IPF_RGB.
- static Image * FromBuffer (unsigned char *raw_data, int raw_data_length, int width, int height, int stride, int channels)

  Factory method for loading an image from uncompressed pixels buffer, with UINT8 channel container. Copies the buffer internally. Buffers with types IPF_G, IPF_RGB, and IPF_BGRA are assumed.
- static Image * FromBufferExtended (unsigned char *raw_data, int raw_data_length, int width, int height, int stride, ImagePixelFormat pixel_format, int bytes_per_channel)

  Factory method for loading an image from an uncompressed pixel buffer with extended settings. Copies the buffer internally.
- static Image * FromYUVBuffer (unsigned char *yuv_data, int yuv_data_length, int width, int height)

  Factory method for loading an image from YUV NV21 buffer.
- static Image * FromYUV (unsigned char *y_plane, int y_plane_length, unsigned char *u_plane, int u_plane_length, unsigned char *v_plane, int v_plane_length, const YUVDimensions &dimensions)

  Factory method for loading an image from a universal YUV buffer.
- static Image * FromBase64Buffer (const char *base64_buffer, const int page_number=0, const Size &max_size=Size(25000, 25000))

  Factory method for loading an image from file pre-loaded in a buffer encoded as a Base64 string. Will be treated as IPF_G or IPF_RGB.

1.4.1 Detailed Description

Class representing bitmap image.

Definition at line 79 of file se_image.h.

1.4.2 Member Function Documentation

GetNumberOfPages()

```
static int se::common::Image::GetNumberOfPages (
            const char * image_filename )  [static]
```

Returns the number of pages in an image.

Parameters

image_filename	path to an imag file

Returns

the number of pages in an image

GetImagePageName()

```
static MutableString se::common::Image::GetImagePageName (
            const char * image_filename,
            int page_number )  [static]
```

Returns the name of the specified page.

Parameters

image_filename	The filename of the image to process.
page_number	0-based page number.

Returns

Separate page filename.

CreateEmpty()

```
static Image * se::common::Image::CreateEmpty ( )  [static]
```

Factory method for creating an empty image.

Returns

Pointer to a created image. New object is allocated, the caller is responsible for deleting it.

FromFile()

```
static Image * se::common::Image::FromFile (
            const char * image_filename,
            const int page_number = 0,
            const Size & max_size = Size(25000, 25000) )  [static]
```

Factory method for loading an image from file. Will be treated as IPF_G or IPF_RGB.

Parameters

image_filename	path to an image file (png, jpg, tif)
page_number	page number (0 by default)
max_size	maximum image size in pixels (0 for unrestricted)

Returns

Pointer to a created image. New object is allocated, the caller is responsible for deleting it.

FromFileBuffer()

```
static Image * se::common::Image::FromFileBuffer (
            unsigned char * data,
            int data_length,
            const int page_number = 0,
            const Size & max_size = Size(25000, 25000) )  [static]
```

Factory method for loading an image from file pre-loaded in a buffer Will be treated as IPF_G or IPF_RGB.

Parameters

data	pointer to a loaded file buffer
data_length	size of the loaded file buffer
page_number	page number (0 by default)
max_size	maximum image size in pixels (0 for unrestricted)

Returns

Pointer to a created image. New object is allocated, the caller is responsible for deleting it.

FromBuffer()

```
static Image * se::common::Image::FromBuffer (
            unsigned char * raw_data,
            int raw_data_length,
            int width,
            int height,
            int stride,
            int channels )  [static]
```

Factory method for loading an image from uncompressed pixels buffer, with UINT8 channel container. Copies the buffer internally. Buffers with types IPF_G, IPF_RGB, and IPF_BGRA are assumed.

Parameters

raw_data	- pointer to a pixels buffer
raw_data_length	size of the pixels buffer
width	width of the image in pixels
height	height of the image in pixels
stride	size of an image row in bytes (including alignment)
channels	number of channels per-pixel

Returns

Pointer to a created image. New object is allocated, the caller is responsible for deleting it.

FromBufferExtended()

```
static Image * se::common::Image::FromBufferExtended (
            unsigned char * raw_data,
            int raw_data_length,
            int width,
            int height,
            int stride,
            ImagePixelFormat pixel_format,
            int bytes_per_channel )  [static]
```

Factory method for loading an image from an uncompressed pixel buffer with extended settings. Copies the buffer internally.

Parameters

raw_data	pointer to a pixels buffer
raw_data_length	size of the pixels buffer
width	width of the image in pixels
height	height of the image in pixels
stride	size of an image row in bytes (including alignment)
pixel_format	pixel format
bytes_per_channel	size of a pixel component in bytes

Returns

Pointer to a created image. New object is allocated, the caller is responsible for deleting it.

FromYUVBuffer()

```
static Image * se::common::Image::FromYUVBuffer (
            unsigned char * yuv_data,
            int yuv_data_length,
            int width,
            int height )  [static]
```

Factory method for loading an image from YUV NV21 buffer.

Parameters

yuv_data	pointer to YUV NV21 buffer
yuv_data_length	size of the YUV NV21 buffer
width	width of the image in pixels
height	height of the image in pixels

Returns

Pointer to a created image. New object is allocated, the caller is responsible for deleting it.

FromYUV()

```
static Image * se::common::Image::FromYUV (
            unsigned char * y_plane,
            int y_plane_length,
            unsigned char * u_plane,
            int u_plane_length,
            unsigned char * v_plane,
            int v_plane_length,
            const YUVDimensions & dimensions )  [static]
```

Factory method for loading an image from a universal YUV buffer.

Parameters

y_plane	pointer to Y plane buffer
y_plane_length	Y plane buffer length
u_plane	pointer to U plane buffer
u_plane_length	U plane buffer length
v_plane	pointer to V plane buffer
v_plane_length	V plane buffer length
dimensions	YUV parameters and dimensions

Returns

Pointer to a created image. New object is allocated, the caller is responsible for deleting it.

FromBase64Buffer()

```
static Image * se::common::Image::FromBase64Buffer (
            const char * base64_buffer,
            const int page_number = 0,
            const Size & max_size = Size(25000, 25000) )  [static]
```

Factory method for loading an image from file pre-loaded in a buffer encoded as a Base64 string. Will be treated as IPF_G or IPF_RGB.

Parameters

base64_buffer	pointer to a base64 file buffer
page_number	page number (0 by default)
max_size	maximum image size in pixels (0 for unrestricted)

Returns

Pointer to a created image. New object is allocated, the caller is responsible for deleting it.

GetNumberOfLayers()

```
virtual int se::common::Image::GetNumberOfLayers ( ) const  [pure virtual]
```

Gets the number of additional layers.

Returns

The number of layers

GetLayer()

```
virtual const Image & se::common::Image::GetLayer (
            const char * name ) const  [pure virtual]
```

Gets the additional layer by the specified name.

Parameters

name	the name of the required layer

Returns

The layer

GetLayerPtr()

```
virtual const Image * se::common::Image::GetLayerPtr (
            const char * name ) const  [pure virtual]
```

Gets the additional layer by the specified name.

Parameters

name	the name of the required layer

Returns

The pointer to the layer

LayersBegin()

```
virtual ImagesMapIterator se::common::Image::LayersBegin ( ) const  [pure virtual]
```

Gets the 'begin' map iterator to the internal layers collection.

Returns

The 'begin' map iterator to the internal layers collection

LayersEnd()

```
virtual ImagesMapIterator se::common::Image::LayersEnd ( ) const  [pure virtual]
```

Gets the 'end' map iterator to the internal layers collection.

Returns

The 'end' map iterator to the internal layers collection

HasLayer()

```
virtual bool se::common::Image::HasLayer (
            const char * name ) const  [pure virtual]
```

Checks whether the Image contains the layer with the specified name.

Parameters

name	the name of the required layer

Returns

whether the Image contains the layer with the specified name

HasLayers()

```
virtual bool se::common::Image::HasLayers ( ) const  [pure virtual]
```

Checks whether the Image contains the layers.

Returns

whether the Image contains the layers

RemoveLayer()

```
virtual void se::common::Image::RemoveLayer (
            const char * name )  [pure virtual]
```

Removes the layer with the specified name.

Parameters

name	the name of the removable layer

SetLayer()

```
virtual void se::common::Image::SetLayer (
            const char * name,
            const Image & image )  [pure virtual]
```

Add the image with the specified name to the internal layers collection with copying of the pixels of the given image.

Parameters

name	the name of the new layer
image	the value of the new layer

SetLayerWithOwnership()

```
virtual void se::common::Image::SetLayerWithOwnership (
            const char * name,
            Image * image )  [pure virtual]
```

Add the image with the specified name to the internal layers collection by transfering the given image to the internal layers collection. The caller has to release the ownership of the set image.

Parameters

name	the name of the new layer
image	the pointer to the value of the new layer

CloneDeep()

```
virtual Image * se::common::Image::CloneDeep ( ) const  [pure virtual]
```

Clones an image with copying of all pixels.

Returns

Pointer to a cloned image. New object is allocated, the caller is responsible for deleting it.

CloneShallow()

```
virtual Image * se::common::Image::CloneShallow ( ) const  [pure virtual]
```

Clones an image without copying the pixels. The cloned image will be a separate object without memory ownership, the operations with it will be invalid if the source is deallocated.

Returns

Pointer to a cloned image. New object is allocated, the caller is responsible for deleting it.

GetRequiredBufferLength()

```
virtual int se::common::Image::GetRequiredBufferLength ( ) const  [pure virtual]
```

Gets the required buffer length for copying the image pixels into an external pixels buffer.

Returns

Number of required bytes

CopyToBuffer()

```
virtual int se::common::Image::CopyToBuffer (
            unsigned char * buffer,
            int buffer_length ) const  [pure virtual]
```

Copies the image pixels.

Parameters

buffer	pointer to an output pixels buffer
buffer_length	available buffer size. Must be at least the size returned by the GetRequiredBufferLength() method.

Returns

The number of written bytes

Save()

```
virtual void se::common::Image::Save (
            const char * image_filename ) const  [pure virtual]
```

Saves the image to an external file (png, jpg, tif). Format is deduced from the filename extension.

Parameters

image_filename	filename to save the image

GetRequiredBase64BufferLength()

```
virtual int se::common::Image::GetRequiredBase64BufferLength ( ) const  [pure virtual]
```

Returns required buffer size for Base64 JPEG representation of an image. WARNING: will perform one extra JPEG encoding of an image.

Returns

Buffer size in bytes.

CopyBase64ToBuffer()

```
virtual int se::common::Image::CopyBase64ToBuffer (
            char * out_buffer,
            int buffer_length ) const  [pure virtual]
```

Copies the Base64 JPEG representation of an image to an external buffer.

Parameters

out_buffer	output buffer for Base64 JPEG representation
buffer_length	available buffer size. Must be at least the size return by the GetRequiredBase64BufferLength() method.

Returns

The number of written bytes.

GetBase64String()

```
virtual MutableString se::common::Image::GetBase64String ( ) const  [pure virtual]
```

Returns Base64 JPEG representation of an image.

Returns

Base64 JPEG representation in a MutableString form

EstimateFocusScore()

```
virtual double se::common::Image::EstimateFocusScore (
            double quantile = 0.95 ) const  [pure virtual]
```

Estimates focus score of an image.

Parameters

quantile	the derivatives quantile used to estimate focus score

Returns

Focus score of an image

Resize()

```
virtual void se::common::Image::Resize (
            const Size & new_size )  [pure virtual]
```

Scale the image to a new size.

Parameters

new_size	new size of the image

CloneResized()

```
virtual Image * se::common::Image::CloneResized (
            const Size & new_size ) const  [pure virtual]
```

Clones the image scaled to a new size.

Parameters

new_size	new size of the image

Returns

Pointer to a scaled image. New object is allocated, the caller is responsible for deleting it.

Crop() [1/3]

```
virtual void se::common::Image::Crop (
            const Quadrangle & quad )  [pure virtual]
```

Projectively crops a region of image, with approximate selection of the cropped image size.

Parameters

quad	quadrangle in the image for cropping.

CloneCropped() [1/3]

```
virtual Image * se::common::Image::CloneCropped (
            const Quadrangle & quad ) const  [pure virtual]
```

Clones the image projectively cropped with approximate selection of the target image size.

Parameters

quad	quadrangle in the image for cropping

Returns

Pointer to a cropped image. New object is allocated, the caller is responsible for deleting it.

Crop() [2/3]

```
virtual void se::common::Image::Crop (
            const Quadrangle & quad,
            const Size & size )  [pure virtual]
```

Projectively crops a region of image, with a given target size.

Parameters

quad	quadrangle in the image for cropping
size	target cropped image size

CloneCropped() [2/3]

```
virtual Image * se::common::Image::CloneCropped (
            const Quadrangle & quad,
            const Size & size ) const  [pure virtual]
```

Clones the image projectively cropped with a given target size.

Parameters

quad	quadrangle in the image for cropping
size	target cropped image size

Returns

Pointer to a cropped image. New object is allocated, the caller is responsible for deleting it.

Crop() [3/3]

```
virtual void se::common::Image::Crop (
            const Rectangle & rect )  [pure virtual]
```

Crops an image to a rectangular image region.

Parameters

rect	rectangular region to crop

CloneCropped() [3/3]

```
virtual Image * se::common::Image::CloneCropped (
            const Rectangle & rect ) const  [pure virtual]
```

Clones the image cropped to a selected rectangular region (with copying of pixels)

Parameters

rect	rectangular region to crop

Returns

Pointer to a cropped image. New object is allocated, the caller is responsible for deleting it.

CloneCroppedShallow()

```
virtual Image * se::common::Image::CloneCroppedShallow (
            const Rectangle & rect ) const  [pure virtual]
```

Clones the image cropped to a selected rectangular region, without copying of pixels. The cloned image will be a separate object without memory ownership, the operations with it will be invalid if the source is deallocated.

Parameters

rect	rectangular region to crop

Returns

Pointer to a cropped image. New object is allocated, the caller is responsible for deleting it.

Mask() [1/2]

```
virtual void se::common::Image::Mask (
            const Rectangle & rect,
            int pixel_expand = 0,
            double pixel_density = 0 )  [pure virtual]
```

Masks image region specified by rectangle.

Parameters

rect	rectangle region to mask
pixel_expand	expand offset in pixels for each point (0 by default)
pixel_density	reduce dencity of pixels (0 by default)

CloneMasked() [1/2]

```
virtual Image * se::common::Image::CloneMasked (
            const Rectangle & rect,
            int pixel_expand = 0 ) const  [pure virtual]
```

Clone the image with masked region specified by rectangle.

Parameters

rect	rectangle region to mask
pixel_expand	expand offset in pixels for each point (0 by default)

Returns

Pointer to a masked image. New object is allocated, the caller is responsible for deleting it.

Mask() [2/2]

```
virtual void se::common::Image::Mask (
            const Quadrangle & quad,
            int pixel_expand = 0,
            double pixel_density = 0 )  [pure virtual]
```

Mask image region specified by quadrangle.

Parameters

quad	quadrangle region to mask
pixel_expand	expand offset in pixels for each point (0 by default)

CloneMasked() [2/2]

```
virtual Image * se::common::Image::CloneMasked (
            const Quadrangle & quad,
            int pixel_expand = 0 ) const  [pure virtual]
```

Clone the image with masked region specified by quadrangle.

Parameters

quad	quadrangle region to mask
pixel_expand	expand offset in pixels for each point (0 by default)
pixel_density	reduce dencity of pixels (0 by default)

Returns

Pointer to a masked image. New object is allocated, the caller is responsible for deleting it.

Fill() [1/2]

```
virtual void se::common::Image::Fill (
            const Rectangle & rect,
            int ch1,
            int ch2 = 0,
            int ch3 = 0,
            int ch4 = 0,
            int pixel_expand = 0 )  [pure virtual]
```

Fills image region specified by rectangle and color. The method will use the first as many channel values as there are channels in the image.

Parameters

rect	rectangle region to fill
ch1	1-st channel value
ch2	2-nd channel value
ch3	3-rd channel value
ch4	4-th channel value
pixel_expand	expand offset in pixels for each point (0 by default)

CloneFilled() [1/2]

```
virtual Image * se::common::Image::CloneFilled (
            const Rectangle & rect,
            int ch1,
            int ch2 = 0,
            int ch3 = 0,
            int ch4 = 0,
            int pixel_expand = 0 ) const  [pure virtual]
```

Clone the image with filled region specified by rectangle and color. The method will use the first as many channel values as there are channels in the image.

Parameters

rect	rectangle region to fill
ch1	1-st channel value
ch2	2-nd channel value
ch3	3-rd channel value
ch4	4-th channel value
pixel_expand	expand offset in pixels for each point (0 by default)

Returns

Pointer to a filled image. New object is allocated, the caller is responsible for deleting it.

Fill() [2/2]

```
virtual void se::common::Image::Fill (
            const Quadrangle & quad,
```

```
            int ch1,
            int ch2 = 0,
            int ch3 = 0,
            int ch4 = 0,
            int pixel_expand = 0 )  [pure virtual]
```

Fill image region specified by quadrangle and color. The method will use the first as many channel values as there are channels in the image.

Parameters

quad	quadrangle region to fill
ch1	1-st channel value
ch2	2-nd channel value
ch3	3-rd channel value
ch4	4-th channel value
pixel_expand	expand offset in pixels for each point (0 by default)

CloneFilled() [2/2]

```
virtual Image * se::common::Image::CloneFilled (
            const Quadrangle & quad,
            int ch1,
            int ch2 = 0,
            int ch3 = 0,
            int ch4 = 0,
            int pixel_expand = 0 ) const  [pure virtual]
```

Clone the image with filled region specified by quadrangle and color. The method will use the first as many channel values as there are channels in the image.

Parameters

quad	quadrangle region to fill
ch1	1-st channel value
ch2	2-nd channel value
ch3	3-rd channel value
ch4	4-th channel value
pixel_expand	expand offset in pixels for each point (0 by default)

Returns

Pointer to a filled image. New object is allocated, the caller is responsible for deleting it.

CloneFlippedVertical()

```
virtual Image * se::common::Image::CloneFlippedVertical ( ) const  [pure virtual]
```

Clones the image flipped around the vertical axis.

Returns

Pointer to a flipped image. New object is allocated, the caller is responsible for deleting it.

CloneFlippedHorizontal()

```
virtual Image * se::common::Image::CloneFlippedHorizontal ( ) const  [pure virtual]
```

Clones the image flipped around the horizontal axis.

Returns

Pointer to a flipped image. New object is allocated, the caller is responsible for deleting it.

Rotate90()

```
virtual void se::common::Image::Rotate90 (
            int times )  [pure virtual]
```

Rotates the image clockwise by a multiple of 90 degrees.

Parameters

times	the number of times to rotate

CloneRotated90()

```
virtual Image * se::common::Image::CloneRotated90 (
            int times ) const  [pure virtual]
```

Clones the image rotated clockwise by a multiple of 90 degrees.

Parameters

times	the number of times to rotate

Returns

Pointer to a rotated image. New object is allocated, the caller is responsible for deleting it.

CloneAveragedChannels()

```
virtual Image * se::common::Image::CloneAveragedChannels ( ) const  [pure virtual]
```

Clones the image with averaged channel intensity values.

Returns

Pointer to a created image. New object is allocated, the caller is responsible for deleting it.

CloneInverted()

```
virtual Image * se::common::Image::CloneInverted ( ) const  [pure virtual]
```

Clones the image with inverted colos.

Returns

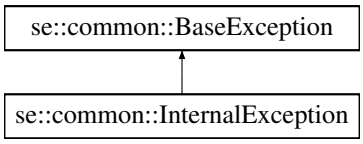
Pointer to a created image. New object is allocated, the caller is responsible for deleting it

1.5 se::common::InternalException Class Reference

InternalException: thrown if an unknown error occurs or if the error occurs within internal system components.

```
#include <se_exception.h>
```

Inheritance diagram for se::common::InternalException:



Public Member Functions

- **InternalException** (const char *msg)
  Ctor with an exception message.
- **InternalException** (const InternalException &copy)
  Copy ctor.
- virtual ~**InternalException** () override=default
  Default dtor.
- virtual const char * ExceptionName () const override
  Returns exception class name.

Public Member Functions inherited from se::common::BaseException

- virtual ~**BaseException** ()
  Non-trivial dtor.
- **BaseException** (const BaseException &copy)
  Copy ctor.
- virtual const char * **what** () const
  Returns exception message.

Additional Inherited Members

Protected Member Functions inherited from se::common::BaseException

- **BaseException** (const char *msg)
  Protected ctor.

1.5.1 Detailed Description

InternalException: thrown if an unknown error occurs or if the error occurs within internal system components.

Definition at line 192 of file se_exception.h.

1.5.2 Member Function Documentation

ExceptionName()

```
virtual const char * se::common::InternalException::ExceptionName ( ) const  [override], [virtual]
```

Returns exception class name.

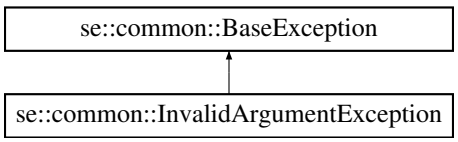
Reimplemented from se::common::BaseException.

1.6 se::common::InvalidArgumentException Class Reference

InvalidArgumentException: thrown if a method is called with invalid input parameters.

```
#include <se_exception.h>
```

Inheritance diagram for se::common::InvalidArgumentException:



Public Member Functions

- **InvalidArgumentException** (const char *msg)
  Ctor with an exception message.
- **InvalidArgumentException** (const InvalidArgumentException &copy)
  Copy ctor.
- virtual ~**InvalidArgumentException** () override=default
  Default dtor.
- virtual const char * ExceptionName () const override
  Returns exception class name.

Public Member Functions inherited from se::common::BaseException

- virtual ~**BaseException** ()
  Non-trivial dtor.
- **BaseException** (const BaseException &copy)
  Copy ctor.
- virtual const char * **what** () const
  Returns exception message.

Additional Inherited Members

Protected Member Functions inherited from se::common::BaseException

- **BaseException** (const char ∗msg)

  Protected ctor.

1.6.1 Detailed Description

InvalidArgumentException: thrown if a method is called with invalid input parameters.

Definition at line 132 of file se_exception.h.

1.6.2 Member Function Documentation

ExceptionName()

```
virtual const char * se::common::InvalidArgumentException::ExceptionName ( ) const  [override],
[virtual]
```

Returns exception class name.

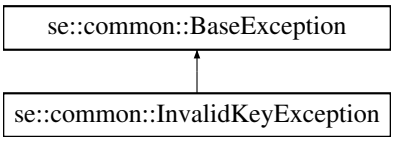
Reimplemented from se::common::BaseException.

1.7 se::common::InvalidKeyException Class Reference

InvalidKeyException: thrown if to an associative container the access is performed with an invalid or a non-existent key, or if the access to a list is performed with an invalid or out-of-range index.

```
#include <se_exception.h>
```

Inheritance diagram for se::common::InvalidKeyException:



Public Member Functions

- **InvalidKeyException** (const char ∗msg)

  Ctor with an exception message.
- **InvalidKeyException** (const InvalidKeyException &copy)

  Copy ctor.
- virtual ∼**InvalidKeyException** () override=default

  Default dtor.
- virtual const char ∗ ExceptionName () const override

  Returns exception class name.

Public Member Functions inherited from se::common::BaseException

- virtual ~**BaseException** ()
  Non-trivial dtor.
- **BaseException** (const BaseException &copy)
  Copy ctor.
- virtual const char * **what** () const
  Returns exception message.

Additional Inherited Members

Protected Member Functions inherited from se::common::BaseException

- **BaseException** (const char *msg)
  Protected ctor.

1.7.1 Detailed Description

InvalidKeyException: thrown if to an associative container the access is performed with an invalid or a non-existent key, or if the access to a list is performed with an invalid or out-of-range index.

Definition at line 50 of file se_exception.h.

1.7.2 Member Function Documentation

ExceptionName()

```
virtual const char * se::common::InvalidKeyException::ExceptionName ( ) const  [override],
[virtual]
```

Returns exception class name.

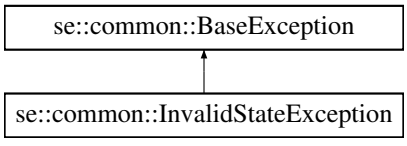
Reimplemented from se::common::BaseException.

1.8 se::common::InvalidStateException Class Reference

InvalidStateException: thrown if an error occurs within the system in relation to an incorrect internal state of the system objects.

```
#include <se_exception.h>
```

Inheritance diagram for se::common::InvalidStateException:



Public Member Functions

- **InvalidStateException** (const char ∗msg)

  Ctor with an exception message.
- **InvalidStateException** (const InvalidStateException &copy)

  Copy ctor.
- virtual ∼**InvalidStateException** () override=default

  Default dtor.
- virtual const char ∗ ExceptionName () const override

  Returns exception class name.

Public Member Functions inherited from se::common::BaseException

- virtual ∼**BaseException** ()

  Non-trivial dtor.
- **BaseException** (const BaseException &copy)

  Copy ctor.
- virtual const char ∗ **what** () const

  Returns exception message.

Additional Inherited Members

Protected Member Functions inherited from se::common::BaseException

- **BaseException** (const char ∗msg)

  Protected ctor.

1.8.1 Detailed Description

InvalidStateException: thrown if an error occurs within the system in relation to an incorrect internal state of the system objects.

Definition at line 172 of file se_exception.h.

1.8.2 Member Function Documentation

ExceptionName()

```
virtual const char * se::common::InvalidStateException::ExceptionName ( ) const  [override],
[virtual]
```

Returns exception class name.

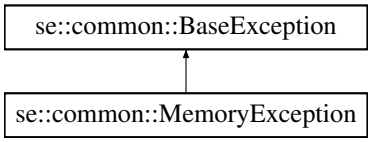
Reimplemented from se::common::BaseException.

1.9 se::common::MemoryException Class Reference

MemoryException: thrown if an allocation is attempted with insufficient RAM.

`#include <se_exception.h>`

Inheritance diagram for se::common::MemoryException:



Public Member Functions

- **MemoryException** (const char *msg)
  Ctor with an exception message.
- **MemoryException** (const MemoryException &copy)
  Copy ctor.
- virtual ~**MemoryException** () override=default
  Default dtor.
- virtual const char * ExceptionName () const override
  Returns exception class name.

Public Member Functions inherited from se::common::BaseException

- virtual ~**BaseException** ()
  Non-trivial dtor.
- **BaseException** (const BaseException &copy)
  Copy ctor.
- virtual const char * **what** () const
  Returns exception message.

Additional Inherited Members

Protected Member Functions inherited from se::common::BaseException

- **BaseException** (const char *msg)
  Protected ctor.

1.9.1 Detailed Description

MemoryException: thrown if an allocation is attempted with insufficient RAM.

Definition at line 152 of file se_exception.h.

1.9.2 Member Function Documentation

ExceptionName()

```
virtual const char * se::common::MemoryException::ExceptionName ( ) const  [override], [virtual]
```

Returns exception class name.

Reimplemented from se::common::BaseException.

1.10 se::common::MutableString Class Reference

Class representing a mutable, memory-owner string.

```
#include <se_string.h>
```

Public Member Functions

- **MutableString** ()
  Default ctor, creates an empty string.
- **MutableString** (const char *c_str)
  Ctor from a C-string.
- **MutableString** (const MutableString &other)
  Copy ctor.
- MutableString & **operator=** (const MutableString &other)
  Assignment operator.
- ~**MutableString** ()
  Non-trivial dtor.
- MutableString & **operator+=** (const MutableString &other)
  Appends a string to this instance.
- MutableString **operator+** (const MutableString &other) const
  Creates a concatenation of this instance and the other string.
- const char * **GetCStr** () const
  Returns an internal C-string.
- int **GetLength** () const
  Returns the length of the string. WARNING: returns the number of bytes, not the number of UTF-8 characters.
- void **Serialize** (Serializer &serializer) const
  Serializes the string given a serializer object.
- void **SerializeImpl** (SerializerImplBase &serializer_impl) const
  Internal serialization implementation.

Private Attributes

- int len_
  length of the internal string in bytes
- char * buf_
  internal C-string

1.10.1 Detailed Description

Class representing a mutable, memory-owner string.

Definition at line 25 of file se_string.h.

1.10.2 Member Data Documentation

len_

```
int se::common::MutableString::len_  [private]
```

length of the internal string in bytes

Definition at line 62 of file se_string.h.

buf_

```
char* se::common::MutableString::buf_  [private]
```

internal C-string

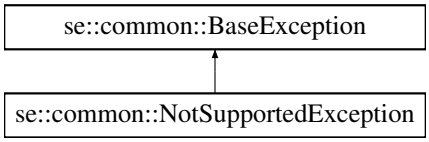
Definition at line 63 of file se_string.h.

1.11 se::common::NotSupportedException Class Reference

NotSupportedException: thrown when trying to access a method which given the current state or given the passed arguments is not supported in the current version of the library or is not supported at all by design.

```
#include <se_exception.h>
```

Inheritance diagram for se::common::NotSupportedException:



Public Member Functions

- **NotSupportedException** (const char *msg)
  Ctor with an exception message.
- **NotSupportedException** (const NotSupportedException &copy)
  Copy ctor.
- virtual ~**NotSupportedException** () override=default
  Default dtor.
- virtual const char * ExceptionName () const override
  Returns exception class name.

Public Member Functions inherited from se::common::BaseException

- virtual ~**BaseException** ()

  Non-trivial dtor.
- **BaseException** (const BaseException &copy)

  Copy ctor.
- virtual const char * **what** () const

  Returns exception message.

Additional Inherited Members

Protected Member Functions inherited from se::common::BaseException

- **BaseException** (const char *msg)

  Protected ctor.

1.11.1 Detailed Description

NotSupportedException: thrown when trying to access a method which given the current state or given the passed arguments is not supported in the current version of the library or is not supported at all by design.

Definition at line 72 of file se_exception.h.

1.11.2 Member Function Documentation

ExceptionName()

```
virtual const char * se::common::NotSupportedException::ExceptionName ( ) const  [override],
[virtual]
```

Returns exception class name.

Reimplemented from se::common::BaseException.

1.12 se::common::OcrChar Class Reference

Class representing an OCR information for a given recognized character.

```
#include <se_string.h>
```

Public Member Functions

- **OcrChar** ()

  Default ctor, creates an empty recognized character.
- OcrChar (const OcrCharVariant *variants, int variants_count, bool is_highlighted, const Quadrangle &quad)

  Main ctor from an array of variants.
- **OcrChar** (const OcrChar &other)

  Copy ctor.
- OcrChar & **operator=** (const OcrChar &other)

  Assignment operator.
- ~**OcrChar** ()

  Non-trivial dtor.
- int **GetVariantsCount** () const

  Gets the number of variants.
- const OcrCharVariant * **GetVariants** () const

  Gets the pointer to the variants array.
- OcrCharVariant & **operator[ ]** (int index)

  Returns the variant by its index (mutable ref)
- const OcrCharVariant & **operator[ ]** (int index) const

  Returns the variant by its index (const ref)
- const OcrCharVariant & **GetVariant** (int index) const

  Returns the variant by its index (const ref)
- OcrCharVariant & **GetMutableVariant** (int index)

  Returns the variant by its index (mutable ref)
- void **SetVariant** (int index, const OcrCharVariant &v)

  Sets the variant to an array with a given index.
- void **Resize** (int size)

  Resizes the variants array to a given size.
- bool **GetIsHighlighted** () const

  Returns the value of the highlight flag.
- void **SetIsHighlighted** (bool is_highlighted)

  Sets the value of the highlight flag.
- const Quadrangle & **GetQuadrangle** () const

  Returns the quadrangle of the OcrChar (const ref)
- Quadrangle & **GetMutableQuadrangle** ()

  Returns the quadrangle of the OcrChar (mutable ref)
- void **SetQuadrangle** (const Quadrangle &quad)

  Sets the quadrangle of the OcrChar.
- void **SortVariants** ()

  Sorts the variants array in the descending order of confidence values.
- const OcrCharVariant & **GetFirstVariant** () const

  Gets the first variant of the array (const ref)
- void **Serialize** (Serializer &serializer) const

  Serializes the object given serializer.
- void **SerializeImpl** (SerializerImplBase &serializer_impl) const

  Internal serialization implementation.

Private Attributes

- int vars_cnt_

  number of variants
- OcrCharVariant * vars_

  variants array
- bool is_highlighted_

  highlight flag
- Quadrangle quad_

  OcrChar quadrangle.

1.12.1 Detailed Description

Class representing an OCR information for a given recognized character.

Definition at line 196 of file se_string.h.

1.12.2 Constructor & Destructor Documentation

OcrChar()

```
se::common::OcrChar::OcrChar (
            const OcrCharVariant * variants,
            int variants_count,
            bool is_highlighted,
            const Quadrangle & quad )
```

Main ctor from an array of variants.

Parameters

variants	pointer to an array of variants
variants_count	the number of variants in the array
is_highlighted	highlight flag for the OcrChar
quad	quadrangle of the OcrChar

1.12.3 Member Data Documentation

vars_cnt_

```
int se::common::OcrChar::vars_cnt_  [private]
```

number of variants

Definition at line 274 of file se_string.h.

vars_

```
OcrCharVariant* se::common::OcrChar::vars_  [private]
```

variants array

Definition at line 275 of file se_string.h.

is_highlighted_

```
bool se::common::OcrChar::is_highlighted_  [private]
```

highlight flag

Definition at line 276 of file se_string.h.

quad_

```
Quadrangle se::common::OcrChar::quad_  [private]
```

OcrChar quadrangle.

Definition at line 277 of file se_string.h.

1.13 se::common::OcrCharVariant Class Reference

Class representing a possible character recognition result.

```
#include <se_string.h>
```

Public Member Functions

- **OcrCharVariant** ()
  Default ctor, creates an empty variant with zero confidence.
- OcrCharVariant (const MutableString &utf8_char, float confidence)
  Ctor from utf8-char represented as a mutable string.
- OcrCharVariant (const char *utf8_char, float confidence)
  Ctor from utf8-char represented as a C-string.
- ~**OcrCharVariant** ()=default
  Default dtor.
- const char * **GetCharacter** () const
  Gets the character as a C-string.
- void **SetCharacter** (const MutableString &utf8_char)
  Sets a character given a MutableString.
- void **SetCharacter** (const char *utf8_char)
  Sets a character given a C-string.
- float **GetConfidence** () const
  Gets the confidence value.
- void **SetConfidence** (float confidence)
  Sets the confidence value (must be in range [0, 1])
- float **GetInternalScore** () const
  Returns the internal score of the OcrCharVariant.
- void **SetInternalScore** (float internal_score)
  Sets the internal score of the OcrCharVariant.
- void **Serialize** (Serializer &serializer) const
  Serializes the object given a serializer.
- void **SerializeImpl** (SerializerImplBase &serializer_impl) const
  Internal serialization implementation.

Private Attributes

- MutableString char_

  character recognition result representation
- float conf_

  confidence value
- float internal_score_

  internal score

1.13.1 Detailed Description

Class representing a possible character recognition result.

Definition at line 70 of file se_string.h.

1.13.2 Constructor & Destructor Documentation

OcrCharVariant() [1/2]

```
se::common::OcrCharVariant::OcrCharVariant (
            const MutableString & utf8_char,
            float confidence )
```

Ctor from utf8-char represented as a mutable string.

Parameters

utf8_char	utf8-character represented as a mutable string
confidence	float confidence in range [0, 1]

OcrCharVariant() [2/2]

```
se::common::OcrCharVariant::OcrCharVariant (
            const char * utf8_char,
            float confidence )
```

Ctor from utf8-char represented as a C-string.

Parameters

utf8_char	utf8-character represented as a C-string
confidence	float confidence in range [0, 1]

1.13.3 Member Data Documentation

char_

`MutableString se::common::OcrCharVariant::char_ [private]`

character recognition result representation

Definition at line 120 of file se_string.h.

conf_

`float se::common::OcrCharVariant::conf_ [private]`

confidence value

Definition at line 121 of file se_string.h.

internal_score_

`float se::common::OcrCharVariant::internal_score_ [private]`

internal score

Definition at line 122 of file se_string.h.

1.14 se::common::OcrPair Class Reference

Public Member Functions

- **OcrPair** ()
  Default ctor, creates an empty variant with zero confidence.
- OcrPair (const MutableString &utf8_char, float confidence)
  Ctor from utf8-char represented as a mutable string.
- OcrPair (const char *utf8_char, float confidence)
  Ctor from utf8-char represented as a C-string.
- OcrPair (const MutableString &utf8_char, int confidence)
  Ctor from utf8-char represented as a mutable string.
- OcrPair (const char *utf8_char, int confidence)
  Ctor from utf8-char represented as a C-string.
- ~**OcrPair** ()=default
  Default dtor.
- const char * **GetCharacter** () const
  Gets the character as a C-string.
- void **SetCharacter** (const MutableString &utf8_char)
  Sets a character given a MutableString.
- void **SetCharacter** (const char *utf8_char)
  Sets a character given a C-string.
- unsigned char **GetConfidence** () const
  Gets the confidence value.
- void **SetConfidence** (float confidence)
  Sets the confidence float value must be in range [0, 1].
- void **SetConfidence** (int confidence)
  Sets the confidence int value must be in range [0, 255].
- void **Serialize** (Serializer &serializer) const
  Serializes the object given a serializer.
- void **SerializeImpl** (SerializerImplBase &serializer_impl) const
  Internal serialization implementation.

Private Attributes

- MutableString char_

  character recognition result representation
- unsigned char conf_

  confidence value

1.14.1 Detailed Description

Definition at line 127 of file se_string.h.

1.14.2 Constructor & Destructor Documentation

OcrPair() [1/4]

```
se::common::OcrPair::OcrPair (
            const MutableString & utf8_char,
            float confidence )
```

Ctor from utf8-char represented as a mutable string.

Parameters

utf8_char	utf8-character represented as a mutable string
confidence	float confidence in range [0, 1], converted to [0, 255]

OcrPair() [2/4]

```
se::common::OcrPair::OcrPair (
            const char * utf8_char,
            float confidence )
```

Ctor from utf8-char represented as a C-string.

Parameters

utf8_char	utf8-character represented as a C-string
confidence	float confidence in range [0, 1], converted to [0, 255]

OcrPair() [3/4]

```
se::common::OcrPair::OcrPair (
            const MutableString & utf8_char,
            int confidence )
```

Ctor from utf8-char represented as a mutable string.

Parameters

utf8_char	utf8-character represented as a mutable string
confidence	int confidence in range [0, 255]

OcrPair() [4/4]

```
se::common::OcrPair::OcrPair (
            const char * utf8_char,
            int confidence )
```

Ctor from utf8-char represented as a C-string.

Parameters

utf8_char	utf8-character represented as a C-string
confidence	int confidence in range [0, 255]

1.14.3 Member Data Documentation

char_

```
MutableString se::common::OcrPair::char_  [private]
```

character recognition result representation

Definition at line 188 of file se_string.h.

conf_

```
unsigned char se::common::OcrPair::conf_  [private]
```

confidence value

Definition at line 189 of file se_string.h.

1.15 se::common::OcrString Class Reference

Class representing text string recognition result.

```
#include <se_string.h>
```

Public Member Functions

- **OcrString** ()

  Default ctor.
- OcrString (const char *utf8_str)

  Ctor from utf8 C-string. Splits the utf8-string into utf8-characters and creates an OcrChar for each one.
- OcrString (const OcrChar *chars, int chars_count)

  Ctor from an array of characters.
- **OcrString** (const OcrString &other)

  Copy ctor.
- OcrString & **operator=** (const OcrString &other)

  Assignment operator.
- ~**OcrString** ()

  Non-trivial destructor.
- const class OcrStringImpl * **GetOcrStringImplPtr** () const

  Gets the ptr to the OcrStringImpl class (const ptr)
- int **GetCharsCount** () const

  Gets the number of characters.
- const OcrChar * **GetChars** () const

  Gets the pointer to the characters array.
- OcrChar & **operator[ ]** (int index)

  Gets a character by index (mutable ref)
- const OcrChar & **operator[ ]** (int index) const

  Gets a character by index (const ref)
- const OcrChar & **GetChar** (int index) const

  Gets a character by index (const ref)
- OcrChar & **GetMutableChar** (int index)

  Gets a character by index (mutable ref)
- void **SetChar** (int index, const OcrChar &chr)

  Sets a character by index.
- void **AppendChar** (const OcrChar &chr)

  Appends a character.
- void **AppendString** (const OcrString &str)

  Appends a string.
- void **Resize** (int size)

  Resizes the internal array of characters.
- const Quadrangle **GetQuadrangleByIndex** (int idx) const

  Returns the quadrangle of the OcrChar.
- bool **GetAcceptFlag** () const

  Returns accept flag of OcrString.
- bool **GetCellAcceptFlagByIndex** (int idx) const

  Returns accept flag of OcrChar at given idx.
- float **GetBestVariantConfidenceByIndex** (int idx) const

  Returns the confidence of the best OcrCharVariant.
- void **SortVariants** ()

  Sorts the variants in each character by the descending order of confidence.
- MutableString **GetFirstString** () const

  Returns a string composed of the best variants from each OcrChar.
- void **UnpackChars** ()

  Unpack se::common::OcrChars from se::common::OcrString.
- void **RepackChars** ()

Repack se::common::OcrChars to se::common::OcrString.

- void **Serialize** (Serializer &serializer) const

  Serializes the object given serializer.
- void **SerializeImpl** (SerializerImplBase &serializer_impl) const

  Internal serialization implementation.
- OcrPair * **GetPairTable** ()

  Returns top-8 OcrPairs per cell with 8-bit (0–255) confidences.

Static Public Member Functions

- static OcrString ConstructFromImpl (const class OcrStringImpl &ocr_string_impl)

  Ctor from a ptr to OcrStringImpl class.

Private Member Functions

- **OcrString** (const OcrStringImpl &ocr_string_impl)

  Private ctor from an internal implementation structure.

Private Attributes

- OcrStringImpl * ocr_string_impl_

1.15.1 Detailed Description

Class representing text string recognition result.

Definition at line 287 of file se_string.h.

1.15.2 Constructor & Destructor Documentation

OcrString() [1/2]

```
se::common::OcrString::OcrString (
            const char * utf8_str )
```

Ctor from utf8 C-string. Splits the utf8-string into utf8-characters and creates an OcrChar for each one.

Parameters

utf8_str	input utf8 C-string

OcrString() [2/2]

```
se::common::OcrString::OcrString (
            const OcrChar * chars,
            int chars_count )
```

Ctor from an array of characters.

Parameters

chars	array of OcrChars
chars_count	the number of characters

1.15.3 Member Function Documentation

ConstructFromImpl()

```
static OcrString se::common::OcrString::ConstructFromImpl (
            const class OcrStringImpl & ocr_string_impl )  [static]
```

Ctor from a ptr to OcrStringImpl class.

Parameters

ocr_string_impl	ptr to OcrStringImpl class

1.15.4 Member Data Documentation

ocr_string_impl_

```
OcrStringImpl* se::common::OcrString::ocr_string_impl_  [private]
```

Definition at line 391 of file se_string.h.

1.16 se::common::Point Class Reference

Class representing a point in an image.

```
#include <se_geometry.h>
```

Public Member Functions

- **Point** ()

  Default ctor - initializes a point with zero-valued coordinates.
- **Point** (double x, double y)

  Main ctor - initializes both coordinates.
- void **Serialize** (Serializer &serializer) const

  Serialize point given serializer object.
- void **SerializeImpl** (SerializerImplBase &serializer_impl) const

  Internal serialization implementation.

Public Attributes

- double x

  X-coordinate of the point (in pixels)
- double y

  Y-coordinate of the point (in pixels)

1.16.1 Detailed Description

Class representing a point in an image.

Definition at line 47 of file se_geometry.h.

1.16.2 Member Data Documentation

x

```
double se::common::Point::x
```

X-coordinate of the point (in pixels)

Definition at line 62 of file se_geometry.h.

y

```
double se::common::Point::y
```

Y-coordinate of the point (in pixels)

Definition at line 63 of file se_geometry.h.

1.17 se::common::Polygon Class Reference

Class representing a polygon in an image.

```
#include <se_geometry.h>
```

Public Member Functions

- **Polygon** ()

  Default ctor - initializes a polygon with no points.
- **Polygon** (const Point *points, int points_count)

  Main ctor - initializes a polygon with points array (points are copied)
- **Polygon** (const Polygon &other)

  Copy ctor - copies all points of the other polygon.
- Polygon & **operator=** (const Polygon &other)

  Assignment operator - copies all points of the other polygon.
- ~**Polygon** ()

  Dtor (non-trivial)
- int **GetPointsCount** () const

  Returns the number of points in the polygon.
- const Point * **GetPoints** () const

  Returns a pointer to the first point in the polygon.
- Point & **operator[ ]** (int index)

  Mutable subscript getter for a point by an index.
- const Point & **operator[ ]** (int index) const

  Subscript getter for a point by an index.
- const Point & **GetPoint** (int index) const

  Getter for a point by an index.
- Point & **GetMutablePoint** (int index)

  Mutable getter for a point by an index.
- void **SetPoint** (int index, const Point &p)

  Setter for a point by an index.
- void **Resize** (int size)

  Resizes in internal array of points. If size is different from the current size, the new array is allocated. Old points are copied, new points are initialized with zero coordinates (if upsized)
- Rectangle **GetBoundingRectangle** () const

  Calculates, creates, and returns a bounding rectangle for the polygon.
- void **Serialize** (Serializer &serializer) const

  Serialize quadrangle given serializer object.
- void **SerializeImpl** (SerializerImplBase &serializer_impl) const

  Internal serialization implementation.

Private Attributes

- int pts_cnt_

  Number of points.
- Point * pts_

  Points array.

1.17.1 Detailed Description

Class representing a polygon in an image.

Definition at line 274 of file se_geometry.h.

1.17.2 Member Data Documentation

pts_cnt_

```
int se::common::Polygon::pts_cnt_  [private]
```

Number of points.

Definition at line 327 of file se_geometry.h.

pts_

```
Point* se::common::Polygon::pts_  [private]
```

Points array.

Definition at line 328 of file se_geometry.h.

1.18 se::common::ProjectiveTransform Class Reference

Class representing projective transformation of a plane.

```
#include <se_geometry.h>
```

Public Types

- using Raw2dArrayType = double[3][3]
  type declaration for internal matrix

Public Member Functions

- virtual ~**ProjectiveTransform** ()=default
  Default dtor.
- virtual ProjectiveTransform * **Clone** () const =0
  Copies transform object.
- virtual Point **TransformPoint** (const Point &p) const =0
  Transforms an input point.
- virtual Quadrangle **TransformQuad** (const Quadrangle &q) const =0
  Transforms an input quadrangle.
- virtual Polygon **TransformPolygon** (const Polygon &poly) const =0
  Transforms an input polygon.
- virtual bool **IsInvertable** () const =0
  Returns true iff the transformation is invertable.
- virtual void **Invert** ()=0
  Inverts the projective transformation.
- virtual ProjectiveTransform * **CloneInverted** () const =0
  Creates a new object with an inverted transformation.
- virtual const Raw2dArrayType & **GetRawCoeffs** () const =0
  Returns internal transformation matrix (constant)
- virtual Raw2dArrayType & **GetMutableRawCoeffs** ()=0
  Returns internal transformation matrix (mutable)
- virtual void **Serialize** (Serializer &serializer) const =0
  Serializes the projective transformation given serializer object.

Static Public Member Functions

- static bool CanCreate (const Quadrangle &src_quad, const Quadrangle &dst_quad)

  Returns true, iff the projective transform can be defined which transforms the quad 'src_quad' to the quad 'dst_quad'.
- static bool CanCreate (const Quadrangle &src_quad, const Size &dst_size)

  Returns true, iff the projective transform can be defined which transforms the quad 'src_quad' to an orthotropic rectangle with size 'dst_size'.
- static ProjectiveTransform * Create ()

  Creates a unit transformation.
- static ProjectiveTransform * Create (const Quadrangle &src_quad, const Quadrangle &dst_quad)

  Creates a transformation which transforms the quad 'src_quad' to the quad 'dst_quad'.
- static ProjectiveTransform * Create (const Quadrangle &src_quad, const Size &dst_size)

  Create a transformation which transforms the quad 'src_quad' to an orthotropic rectangle with size 'dst_size'.
- static ProjectiveTransform * Create (const Raw2dArrayType &coeffs)

  Creates a transformation given raw matrix.

1.18.1 Detailed Description

Class representing projective transformation of a plane.

Definition at line 335 of file se_geometry.h.

1.18.2 Member Typedef Documentation

Raw2dArrayType

```
using se::common::ProjectiveTransform::Raw2dArrayType = double[3][3]
```

type declaration for internal matrix

Definition at line 337 of file se_geometry.h.

1.18.3 Member Function Documentation

CanCreate() [1/2]

```
static bool se::common::ProjectiveTransform::CanCreate (
            const Quadrangle & src_quad,
            const Quadrangle & dst_quad )  [static]
```

Returns true, iff the projective transform can be defined which transforms the quad 'src_quad' to the quad 'dst_quad'.

Parameters

src_quad	transformation source
dst_quad	transformation destination

Returns

true iff such transform can be defined and constructed

CanCreate() [2/2]

```
static bool se::common::ProjectiveTransform::CanCreate (
            const Quadrangle & src_quad,
            const Size & dst_size )  [static]
```

Returns true, iff the projective transform can be defined which transforms the quad 'src_quad' to an orthotropic rectangle with size 'dst_size'.

Parameters

src_quad	transformation source
dst_size	linear sizes of the transformation destionation

Returns

true iff such transform can be defined and constructed

Create() [1/4]

```
static ProjectiveTransform * se::common::ProjectiveTransform::Create ( )  [static]
```

Creates a unit transformation.

Returns

Unit transformation object

Create() [2/4]

```
static ProjectiveTransform * se::common::ProjectiveTransform::Create (
            const Quadrangle & src_quad,
            const Quadrangle & dst_quad )  [static]
```

Creates a transformation which transforms the quad 'src_quad' to the quad 'dst_quad'.

Parameters

src_quad	transformation source
dst_quad	transformation destination

Returns

Created transform

Create() [3/4]

```
static ProjectiveTransform * se::common::ProjectiveTransform::Create (
            const Quadrangle & src_quad,
            const Size & dst_size )  [static]
```

Create a transformation which transforms the quad 'src_quad' to an orthotropic rectangle with size 'dst_size'.

Parameters

src_quad	transformation source
dst_size	linear sizes of the transformation destination

Returns

Created transform

Create() [4/4]

```
static ProjectiveTransform * se::common::ProjectiveTransform::Create (
            const Raw2dArrayType & coeffs )  [static]
```

Creates a transformation given raw matrix.

Parameters

coeffs	transformation matrix

Returns

Created transform

1.19 se::common::Quadrangle Class Reference

Class representing a quadrangle in an image.

```
#include <se_geometry.h>
```

Public Member Functions

- **Quadrangle** ()

  Default ctor - initializes quadrangle with all points pointing to zero.
- **Quadrangle** (const Point &a, const Point &b, const Point &c, const Point &d)

Main ctor - initializes all four points of the quadrangle.

- Point & **operator[ ]** (int index)

  Mutable subscript getter for a point (indices from 0 to 3)
- const Point & **operator[ ]** (int index) const

  Subscript getter for a point (indices from 0 to 3)
- const Point & **GetPoint** (int index) const

  Getter for a point (indices from 0 to 3)
- Point & **GetMutablePoint** (int index)

  Mutable getter for a point (indices from 0 to 3)
- void **SetPoint** (int index, const Point &p)

  Setter for a point (indices from 0 to 3)
- Rectangle **GetBoundingRectangle** () const

  Calculates, creates, and returns a bounding rectangle for the quadrangle.
- void **Serialize** (Serializer &serializer) const

  Serialize rectangle given serializer object.
- void **SerializeImpl** (SerializerImplBase &serializer_impl) const

  Internal serialization implementation.

Private Attributes

- Point pts_ [4]

  Constituent points.

1.19.1 Detailed Description

Class representing a quadrangle in an image.

Definition at line 93 of file se_geometry.h.

1.19.2 Member Data Documentation

pts_

```
Point se::common::Quadrangle::pts_[4]  [private]
```

Constituent points.

Definition at line 126 of file se_geometry.h.

1.20 se::common::QuadranglesMapIterator Class Reference

QuadranglesMapIterator: iterator object for maps of named quadrangles.

```
#include <se_geometry.h>
```

Public Member Functions

- **QuadranglesMapIterator** (const QuadranglesMapIterator &other)

  Copy ctor.
- QuadranglesMapIterator & **operator=** (const QuadranglesMapIterator &other)

  Assignment operator.
- ~**QuadranglesMapIterator** ()

  Non-trivial dtor.
- const char * **GetKey** () const

  Returns the name of the quadrangle.
- const Quadrangle & **GetValue** () const

  Returns the target quadrangle.
- bool **Equals** (const QuadranglesMapIterator &rvalue) const

  Returns true iff the rvalue iterator points to the same object.
- bool **operator==** (const QuadranglesMapIterator &rvalue) const

  Returns true iff the rvalue iterator points to the same object.
- bool **operator!=** (const QuadranglesMapIterator &rvalue) const

  Returns true iff the rvalue iterator points to a different object.
- void **Advance** ()

  Points an iterator to the next object a the collection.
- void **operator++** ()

  Points an iterator to the next object a the collection.

Static Public Member Functions

- static QuadranglesMapIterator **ConstructFromImpl** (const QuadranglesMapIteratorImpl &pimpl)

  Construction of the iterator object from internal implementation.

Private Member Functions

- **QuadranglesMapIterator** (const QuadranglesMapIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- class QuadranglesMapIteratorImpl * pimpl_

  Internal implementation.

1.20.1 Detailed Description

QuadranglesMapIterator: iterator object for maps of named quadrangles.

Definition at line 184 of file se_geometry.h.

1.20.2 Member Data Documentation

pimpl_

```
class QuadranglesMapIteratorImpl* se::common::QuadranglesMapIterator::pimpl_  [private]
```

Internal implementation.

Definition at line 225 of file se_geometry.h.

1.21 se::common::QuadranglesVectorIterator Class Reference

QuadranglesVectorIterator: iterator object for vector of quadrangles.

```
#include <se_geometry.h>
```

Public Member Functions

- **QuadranglesVectorIterator** (const QuadranglesVectorIterator &other)
  Copy ctor.
- QuadranglesVectorIterator & **operator=** (const QuadranglesVectorIterator &other)
  Assignment operator.
- ~**QuadranglesVectorIterator** ()
  Non-trivial dtor.
- const Quadrangle & **GetValue** () const
  Returns the target quadrangle.
- bool **Equals** (const QuadranglesVectorIterator &rvalue) const
  Returns true iff the rvalue iterator points to the same object.
- bool **operator==** (const QuadranglesVectorIterator &rvalue) const
  Returns true iff the rvalue iterator points to the same object.
- bool **operator!=** (const QuadranglesVectorIterator &rvalue) const
  Returns true iff the rvalue iterator points to a different object.
- void **Advance** ()
  Points an iterator to the next object a the collection.
- void **operator++** ()
  Points an iterator to the next object a the collection.

Static Public Member Functions

- static QuadranglesVectorIterator **ConstructFromImpl** (const QuadranglesVectorIteratorImpl &pimpl)
  Construction of the iterator object from internal implementation.

Private Member Functions

- **QuadranglesVectorIterator** (const QuadranglesVectorIteratorImpl &pimpl)
  Private ctor from internal implementation.

Private Attributes

- class QuadranglesVectorIteratorImpl * pimpl_

  Internal implementation.

1.21.1 Detailed Description

QuadranglesVectorIterator: iterator object for vector of quadrangles.

Definition at line 136 of file se_geometry.h.

1.21.2 Member Data Documentation

pimpl_

```
class QuadranglesVectorIteratorImpl* se::common::QuadranglesVectorIterator::pimpl_  [private]
```

Internal implementation.

Definition at line 174 of file se_geometry.h.

1.22 se::common::Rectangle Class Reference

Class representing a rectangle in an image.

```
#include <se_geometry.h>
```

Public Member Functions

- **Rectangle** ()

  Default ctor - initializes rectangle with zero-valued fields.
- **Rectangle** (int x, int y, int width, int height)

  Main ctor - initializes all fields of a rectangle.
- void **Serialize** (Serializer &serializer) const

  Serialize rectangle given serializer object.
- void **SerializeImpl** (SerializerImplBase &serializer_impl) const

  Internal serialization implementation.

Public Attributes

- int x

  X-coordinate of the top-left corner (in pixels)
- int y

  Y-coordinate of the top-left corner (in pixels)
- int width

  Width of the rectangle (in pixels)
- int height

  Height of the rectangle (in pixels)

1.22.1 Detailed Description

Class representing a rectangle in an image.

Definition at line 22 of file se_geometry.h.

1.22.2 Member Data Documentation

x

```
int se::common::Rectangle::x
```

X-coordinate of the top-left corner (in pixels)

Definition at line 37 of file se_geometry.h.

y

```
int se::common::Rectangle::y
```

Y-coordinate of the top-left corner (in pixels)

Definition at line 38 of file se_geometry.h.

width

```
int se::common::Rectangle::width
```

Width of the rectangle (in pixels)

Definition at line 39 of file se_geometry.h.

height

```
int se::common::Rectangle::height
```

Height of the rectangle (in pixels)

Definition at line 40 of file se_geometry.h.

1.23 se::common::RectanglesVectorIterator Class Reference

Public Member Functions

- **RectanglesVectorIterator** (const RectanglesVectorIterator &other)

  Copy ctor.
- RectanglesVectorIterator & **operator=** (const RectanglesVectorIterator &other)

  Assignment operator.
- ∼**RectanglesVectorIterator** ()

  Non-trivial dtor.
- const Rectangle & **GetValue** () const

  Returns the target rectangle.
- bool **Equals** (const RectanglesVectorIterator &rvalue) const

  Returns true iff the rvalue iterator points to the same object.
- bool **operator==** (const RectanglesVectorIterator &rvalue) const

  Returns true if the rvalue iterator points to the same object.
- bool **operator!=** (const RectanglesVectorIterator &rvalue) const

  Returns true if the rvalue iterator points to a different object.
- void **Advance** ()

  Points an iterator to the next object a the collection.
- void **operator++** ()

  Points an iterator to the next object a the collection.

Static Public Member Functions

- static RectanglesVectorIterator **ConstructFromImpl** (const RectanglesVectorIteratorImpl &pimpl)

  Construction of the iterator object from internal implementation.

Private Member Functions

- **RectanglesVectorIterator** (const RectanglesVectorIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- class RectanglesVectorIteratorImpl ∗ pimpl_

  Internal implementation.

1.23.1 Detailed Description

Definition at line 230 of file se_geometry.h.

1.23.2 Member Data Documentation

pimpl_

```
class RectanglesVectorIteratorImpl* se::common::RectanglesVectorIterator::pimpl_  [private]
```

Internal implementation.

Definition at line 268 of file se_geometry.h.

1.24 se::common::SerializationParameters Class Reference

Class representing serialization parameters.

`#include <se_serialization.h>`

Public Member Functions

- **SerializationParameters** ()
  Default ctor.
- ~**SerializationParameters** ()
  Default dtor.
- **SerializationParameters** (const SerializationParameters &copy)
  Copy ctor.
- SerializationParameters & **operator=** (const SerializationParameters &other)
  Assignment operator.
- bool HasIgnoredObjectType (const char *object_type) const
  Checks whether the serialization parameters have an ignored object type.
- void AddIgnoredObjectType (const char *object_type)
  Adds an object type to the set of ignored.
- void RemoveIgnoredObjectType (const char *object_type)
  Removes an object type from the set of ignored.
- se::common::StringsSetIterator **IgnoredObjectTypesBegin** () const
  Returns a begin iterator to the set of ignored object types.
- se::common::StringsSetIterator **IgnoredObjectTypesEnd** () const
  Returns an end iterator to the set of ignored object types.
- bool HasIgnoredKey (const char *key) const
  Checks whether the serialization parameters have an ignored key.
- void AddIgnoredKey (const char *key)
  Adds a key to the set of ignored keys.
- void RemoveIgnoredKey (const char *key)
  Removes a key from the set of ignored keys.
- se::common::StringsSetIterator **IgnoredKeysBegin** () const
  Returns a begin iterator to the set of ignored keys.
- se::common::StringsSetIterator **IgnoredKeysEnd** () const
  Returns an end iterator to the set of ignored keys.
- const SerializationParametersImpl & **GetImpl** () const
  Returns an internal implementation structure.

Private Attributes

- SerializationParametersImpl * pimpl_
  pointer to internal implementation

1.24.1 Detailed Description

Class representing serialization parameters.

Definition at line 25 of file se_serialization.h.

1.24.2 Member Function Documentation

HasIgnoredObjectType()

```
bool se::common::SerializationParameters::HasIgnoredObjectType (
            const char * object_type ) const
```

Checks whether the serialization parameters have an ignored object type.

Parameters

object_type	the name of the object type to check

Returns

true iff the object type 'object_type' is ignored

AddIgnoredObjectType()

```
void se::common::SerializationParameters::AddIgnoredObjectType (
            const char * object_type )
```

Adds an object type to the set of ignored.

Parameters

object_type	the name of the object type to add

RemoveIgnoredObjectType()

```
void se::common::SerializationParameters::RemoveIgnoredObjectType (
            const char * object_type )
```

Removes an object type from the set of ignored.

Parameters

object_type	the name of the object type to remove

HasIgnoredKey()

```
bool se::common::SerializationParameters::HasIgnoredKey (
            const char * key ) const
```

Checks whether the serialization parameters have an ignored key.

Parameters

key	the name of the key to check

Returns

true iff the key 'key' is ignored

AddIgnoredKey()

```
void se::common::SerializationParameters::AddIgnoredKey (
            const char * key )
```

Adds a key to the set of ignored keys.

Parameters

key	the name of the key to add

RemoveIgnoredKey()

```
void se::common::SerializationParameters::RemoveIgnoredKey (
            const char * key )
```

Removes a key from the set of ignored keys.

Parameters

key	the name of the key to remove

1.24.3 Member Data Documentation

pimpl_

```
SerializationParametersImpl* se::common::SerializationParameters::pimpl_  [private]
```

pointer to internal implementation

Definition at line 94 of file se_serialization.h.

1.25 se::common::Serializer Class Reference

Class representing the serializer object.

```
#include <se_serialization.h>
```

Public Member Functions

- virtual ~**Serializer** ()=default

  Default dtor.
- virtual void **Reset** ()=0

  Resets the serializer state.
- virtual const char * **GetCStr** () const =0

  Returns the serialized string.
- virtual const char * **SerializerType** () const =0

  Returns the name of the serializer type.

Static Public Member Functions

- static Serializer * CreateJSONSerializer (const SerializationParameters &params)

  Factory method for creating a JSON serializer object.

1.25.1 Detailed Description

Class representing the serializer object.

Definition at line 104 of file se_serialization.h.

1.25.2 Member Function Documentation

CreateJSONSerializer()

```
static Serializer * se::common::Serializer::CreateJSONSerializer (
            const SerializationParameters & params )  [static]
```

Factory method for creating a JSON serializer object.

Parameters

params	serialization parameters

Returns

Pointer to a constructed serializer object. New object is created, the caller is responsible for deleting it.

1.26 se::common::Size Class Reference

Class representing a size of the (rectangular) object.

```
#include <se_geometry.h>
```

Public Member Functions

- **Size** ()

  Default ctor - initializes size with zero-valued fields.
- **Size** (int width, int height)

  Main ctor - initializes all fields.
- void **Serialize** (Serializer &serializer) const

  Serialize size given serializer object.
- void **SerializeImpl** (SerializerImplBase &serializer_impl) const

  Internal serialization implementation.

Public Attributes

- int width

  Width.
- int height

  Height.

1.26.1 Detailed Description

Class representing a size of the (rectangular) object.

Definition at line 70 of file se_geometry.h.

1.26.2 Member Data Documentation

width

```
int se::common::Size::width
```

Width.

Definition at line 85 of file se_geometry.h.

height

```
int se::common::Size::height
```

Height.

Definition at line 86 of file se_geometry.h.

1.27 se::common::StringsMapIterator Class Reference

Iterator to a map from strings to strings.

```
#include <se_strings_iterator.h>
```

Public Member Functions

- **StringsMapIterator** (const StringsMapIterator &other)
  Copy ctor.
- StringsMapIterator & **operator=** (const StringsMapIterator &other)
  Assignment operator.
- ~**StringsMapIterator** ()
  Non-trivial dtor.
- const char * **GetKey** () const
  Gets the string key.
- const char * **GetValue** () const
  Gets the string value.
- bool **Equals** (const StringsMapIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const StringsMapIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const StringsMapIterator &rvalue) const
  Returns true iff this instance and rvalue point to the different objects.
- void **Advance** ()
  Shifts the iterator to the next object.
- void **operator++** ()
  Shifts the iterator to the next object.

Static Public Member Functions

- static StringsMapIterator **ConstructFromImpl** (const StringsMapIteratorImpl &pimpl)
  Constructs the iterator from an internal implementation structure.

Private Member Functions

- **StringsMapIterator** (const StringsMapIteratorImpl &pimpl)
  Private ctor from an internal implementation structure.

Private Attributes

- class StringsMapIteratorImpl * pimpl_
  internal implementation

1.27.1 Detailed Description

Iterator to a map from strings to strings.

Definition at line 124 of file se_strings_iterator.h.

1.27.2 Member Data Documentation

pimpl_

```
class StringsMapIteratorImpl* se::common::StringsMapIterator::pimpl_  [private]
```

internal implementation

Definition at line 165 of file se_strings_iterator.h.

1.28 se::common::StringsSetIterator Class Reference

Iterator to a set-like collection of strings.

```
#include <se_strings_iterator.h>
```

Public Member Functions

- **StringsSetIterator** (const StringsSetIterator &other)
  Copy ctor.
- StringsSetIterator & **operator=** (const StringsSetIterator &other)
  Assignment operator.
- ~**StringsSetIterator** ()
  Non-trivial dtor.
- const char * **GetValue** () const
  Gets the string value.
- bool **Equals** (const StringsSetIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const StringsSetIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const StringsSetIterator &rvalue) const
  Returns true iff this instance and rvalue point to the different objects.
- void **Advance** ()
  Shifts the iterator to the next object.
- void **operator++** ()
  Shifts the iterator to the next object.

Static Public Member Functions

- static StringsSetIterator **ConstructFromImpl** (const StringsSetIteratorImpl &pimpl)
  Constructs the iterator from an internal implementation structure.

Private Member Functions

- **StringsSetIterator** (const StringsSetIteratorImpl &pimpl)
  Private ctor from an internal implementation structure.

Private Attributes

- class StringsSetIteratorImpl * pimpl_
  internal implementation

1.28.1 Detailed Description

Iterator to a set-like collection of strings.

Definition at line 75 of file se_strings_iterator.h.

1.28.2 Member Data Documentation

pimpl_

```
class StringsSetIteratorImpl* se::common::StringsSetIterator::pimpl_  [private]
```

internal implementation

Definition at line 113 of file se_strings_iterator.h.

1.29 se::common::StringsVectorIterator Class Reference

Iterator to a vector-like collection of strings.

```
#include <se_strings_iterator.h>
```

Public Member Functions

- **StringsVectorIterator** (const StringsVectorIterator &other)
  Copy ctor.
- StringsVectorIterator & **operator=** (const StringsVectorIterator &other)
  Assignment operator.
- ~**StringsVectorIterator** ()
  Non-trivial dtor.
- const char * **GetValue** () const
  Gets the string value.
- bool **Equals** (const StringsVectorIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const StringsVectorIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const StringsVectorIterator &rvalue) const
  Returns true iff this instance and rvalue point to the different objects.
- void **Advance** ()
  Shifts the iterator to the next object.
- void **operator++** ()
  Shifts the iterator to the next object.

Static Public Member Functions

- static StringsVectorIterator **ConstructFromImpl** (const StringsVectorIteratorImpl &pimpl)
  Constructs the iterator from an internal implementation structure.

Private Member Functions

- **StringsVectorIterator** (const StringsVectorIteratorImpl &pimpl)
  Private ctor from an internal implementation structure.

Private Attributes

- class StringsVectorIteratorImpl * pimpl_
  internal implementation

1.29.1 Detailed Description

Iterator to a vector-like collection of strings.

Definition at line 26 of file se_strings_iterator.h.

1.29.2 Member Data Documentation

pimpl_

```
class StringsVectorIteratorImpl* se::common::StringsVectorIterator::pimpl_  [private]
```

internal implementation

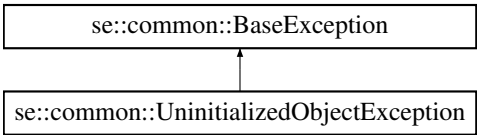
Definition at line 64 of file se_strings_iterator.h.

1.30 se::common::UninitializedObjectException Class Reference

UninitializedObjectException: thrown if an attempt is made to access a non-existent or non-initialized object.

```
#include <se_exception.h>
```

Inheritance diagram for se::common::UninitializedObjectException:



Public Member Functions

- **UninitializedObjectException** (const char *msg)
  Ctor with an exception message.
- **UninitializedObjectException** (const UninitializedObjectException &copy)
  Copy ctor.
- virtual ~**UninitializedObjectException** () override=default
  Default dtor.
- virtual const char * ExceptionName () const override
  Returns exception class name.

Public Member Functions inherited from se::common::BaseException

- virtual ~**BaseException** ()
  Non-trivial dtor.
- **BaseException** (const BaseException &copy)
  Copy ctor.
- virtual const char * **what** () const
  Returns exception message.

Additional Inherited Members

Protected Member Functions inherited from se::common::BaseException

- **BaseException** (const char *msg)
  Protected ctor.

1.30.1 Detailed Description

UninitializedObjectException: thrown if an attempt is made to access a non-existent or non-initialized object.

Definition at line 112 of file se_exception.h.

1.30.2 Member Function Documentation

ExceptionName()

```
virtual const char * se::common::UninitializedObjectException::ExceptionName ( ) const  [override],
[virtual]
```

Returns exception class name.

Reimplemented from se::common::BaseException.

1.31 se::common::YUVDimensions Class Reference

The YUVDimensions struct - extended YUV parameters.

```
#include <se_image.h>
```

Public Member Functions

- **YUVDimensions** ()

  Default ctor.
- **YUVDimensions** (int y_pixel_stride, int y_row_stride, int u_pixel_stride, int u_row_stride, int v_pixel_stride, int v_row_stride, int width, int height, YUVType type)

  Main ctor.

Public Attributes

- int y_plane_pixel_stride

  Y plane pixel stride.
- int y_plane_row_stride

  Y plane row stride.
- int u_plane_pixel_stride

  U plane pixel stride.
- int u_plane_row_stride

  U plane row stride.
- int v_plane_pixel_stride

  V plane pixel stride.
- int v_plane_row_stride

  V plane row stride.
- int width

  image width in pixels
- int height

  image height in pixels
- YUVType type

  YUV format type.

1.31.1 Detailed Description

The YUVDimensions struct - extended YUV parameters.

Definition at line 49 of file se_image.h.

1.31.2 Member Data Documentation

y_plane_pixel_stride

```
int se::common::YUVDimensions::y_plane_pixel_stride
```

Y plane pixel stride.

Definition at line 65 of file se_image.h.

y_plane_row_stride

```
int se::common::YUVDimensions::y_plane_row_stride
```

Y plane row stride.

Definition at line 66 of file se_image.h.

u_plane_pixel_stride

```
int se::common::YUVDimensions::u_plane_pixel_stride
```

U plane pixel stride.

Definition at line 67 of file se_image.h.

u_plane_row_stride

```
int se::common::YUVDimensions::u_plane_row_stride
```

U plane row stride.

Definition at line 68 of file se_image.h.

v_plane_pixel_stride

```
int se::common::YUVDimensions::v_plane_pixel_stride
```

V plane pixel stride.

Definition at line 69 of file se_image.h.

v_plane_row_stride

```
int se::common::YUVDimensions::v_plane_row_stride
```

V plane row stride.

Definition at line 70 of file se_image.h.

width

```
int se::common::YUVDimensions::width
```

image width in pixels

Definition at line 71 of file se_image.h.

height

```
int se::common::YUVDimensions::height
```

image height in pixels

Definition at line 72 of file se_image.h.

type

```
YUVType se::common::YUVDimensions::type
```

YUV format type.

Definition at line 73 of file se_image.h.

1.32 se::doc::DocBarcodeField Class Reference

The class representing a barcode field of a document.

```
#include <doc_fields.h>
```

Public Member Functions

- virtual ~**DocBarcodeField** ()=default
  Default dtor.
- virtual const DocBaseFieldInfo & **GetBaseFieldInfo** () const =0
  Returns the basic field information (const ref)
- virtual DocBaseFieldInfo & **GetMutableBaseFieldInfo** ()=0
  Returns the basic field information (mutable ref)
- virtual const DocBaseFieldInfo * **GetBaseFieldInfoPtr** () const =0
  Returns the basic field information (const ptr)
- virtual DocBaseFieldInfo * **GetMutableBaseFieldInfoPtr** ()=0
  Returns the basic field information (mutable ptr)
- virtual const se::common::MutableString & **GetDecodedString** () const =0
  Returns the barcode decoded message (const ref)
- virtual se::common::MutableString & **GetMutableDecodedString** ()=0
  Returns the barcode decoded message (mutable ref)
- virtual const se::common::MutableString * **GetDecodedStringPtr** () const =0
  Returns the barcode decoded message (const ptr)
- virtual se::common::MutableString * **GetMutableDecodedStringPtr** ()=0
  Returns the barcode decoded message (mutable ptr)
- virtual void **SetDecodedString** (const se::common::MutableString &decstring)=0
  Sets the barcode decoded message.
- virtual void **Serialize** (se::common::Serializer &serializer) const =0
  Serializes the field instance with a given serializer object.

1.32.1 Detailed Description

The class representing a barcode field of a document.

Definition at line 386 of file doc_fields.h.

1.33 se::doc::DocBarcodeFieldsIterator Class Reference

Const-ref iterator for a collection of barcode fields.

`#include <doc_fields_iterators.h>`

Public Member Functions

- **DocBarcodeFieldsIterator** (const DocBarcodeFieldsIterator &other)
  Copy ctor.
- DocBarcodeFieldsIterator & **operator=** (const DocBarcodeFieldsIterator &other)
  Assignment operator.
- ~**DocBarcodeFieldsIterator** ()
  Non-trivial dtor.
- const char * **GetKey** () const
  Returns the field name (the collection key)
- const DocBarcodeField & **GetField** () const
  Returns the field value (const ref)
- const DocBarcodeField * **GetFieldPtr** () const
  Returns the field value (const ptr)
- void **Advance** ()
  Switches the iterator to point on the next field in its collection.
- void **operator++** ()
  Switches the iterator to point on the next field in its collection.
- bool **Equals** (const DocBarcodeFieldsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same field.
- bool **operator==** (const DocBarcodeFieldsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same field.
- bool **operator!=** (const DocBarcodeFieldsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the different fields.

Static Public Member Functions

- static DocBarcodeFieldsIterator **ConstructFromImpl** (const DocBarcodeFieldsIteratorImpl &pimpl)
  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocBarcodeFieldsIterator** (const DocBarcodeFieldsIteratorImpl &pimpl)
  Private ctor from internal implementation.

Private Attributes

- class DocBarcodeFieldsIteratorImpl * pimpl_

  Pointer to internal implementation.

1.33.1 Detailed Description

Const-ref iterator for a collection of barcode fields.

Definition at line 313 of file doc_fields_iterators.h.

1.33.2 Member Data Documentation

pimpl_

```
class DocBarcodeFieldsIteratorImpl* se::doc::DocBarcodeFieldsIterator::pimpl_  [private]
```

Pointer to internal implementation.

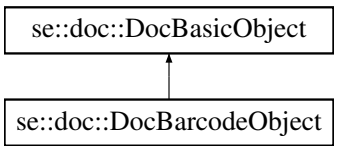
Definition at line 350 of file doc_fields_iterators.h.

1.34 se::doc::DocBarcodeObject Class Reference

The graphical object representing a barcode.

```
#include <doc_objects.h>
```

Inheritance diagram for se::doc::DocBarcodeObject:



Public Member Functions

- virtual ~**DocBarcodeObject** () override=default

  Default dtor.
- virtual const se::common::MutableString & **GetDecodedString** () const =0

  Returns the barcode decoded message (const ref)
- virtual const se::common::MutableString * **GetDecodedStringPtr** () const =0

  Returns the barcode decoded message (const ptr)
- virtual se::common::MutableString & GetMutableDecodedString ()=0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.
- virtual se::common::MutableString * **GetMutableDecodedStringPtr** ()=0

  Returns the barcode decoded message (mutable ptr)
- virtual void **SetDecodedString** (const se::common::MutableString &decstring)=0

  Sets the barcode decoded message.

Public Member Functions inherited from se::doc::DocBasicObject

- virtual ~**DocBasicObject** ()=default

  Default dtor.
- virtual const char * **ObjectType** () const =0

  Returns the name of the concrete object type.
- virtual const DocBaseObjectInfo & **GetBaseObjectInfo** () const =0

  Returns the general basic object info (const ref)
- virtual DocBaseObjectInfo & **GetMutableBaseObjectInfo** ()=0

  Returns the general basic object info (mutable ref ref)
- virtual const DocBaseObjectInfo * **GetBaseObjectInfoPtr** () const =0

  Returns the general basic object info (const ptr)
- virtual DocBaseObjectInfo * **GetMutableBaseObjectInfoPtr** ()=0

  Returns the general basic object info (mutable ptr)
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the object instance with a given serializer object.

Static Public Member Functions

- static const char * **ObjectTypeStatic** ()

  Returns the object type name.

Static Public Member Functions inherited from se::doc::DocBasicObject

- static const char * **BaseClassNameStatic** ()

  Static class name method, returns 'DocBasicObject'.

1.34.1 Detailed Description

The graphical object representing a barcode.

Definition at line 324 of file doc_objects.h.

1.34.2 Member Function Documentation

GetMutableDecodedString()

```
virtual se::common::MutableString & se::doc::DocBarcodeObject::GetMutableDecodedString ( )
[pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Returns the barcode decoded message (mutable ref)

1.35 se::doc::DocBarcodeObjectsCrossPageIterator Class Reference

Basic const-ref iterator for a collection of barcode objects from several pages.

```
#include <doc_physical_document_iterators.h>
```

Public Member Functions

- **DocBarcodeObjectsCrossPageIterator** (const DocBarcodeObjectsCrossPageIterator &other)

  Copy ctor.
- DocBarcodeObjectsCrossPageIterator & **operator=** (const DocBarcodeObjectsCrossPageIterator &other)

  Assignment operator.
- ~**DocBarcodeObjectsCrossPageIterator** ()

  Non-trivial dtor.
- int **GetPhysicalPageID** () const

  Return ID of a physsicak page contaning current object.
- int **GetObjectID** () const

  Return ID of an object.
- const DocBarcodeObject & **GetBarcodeObject** () const

  Returns the barcode object (const ref)
- const DocBarcodeObject * **GetBarcodeObjectPtr** () const

  Returns the barcode object (const ptr)
- void **Advance** ()

  Switches the iterator to point on the next object in its collection.
- bool **Equals** (const DocBarcodeObjectsCrossPageIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocBarcodeObjectsCrossPageIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocBarcodeObjectsCrossPageIterator &rvalue) const

  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocBarcodeObjectsCrossPageIterator **ConstructFromImpl** (const DocBarcodeObjectsCrossPage↩ IteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocBarcodeObjectsCrossPageIterator** (const DocBarcodeObjectsCrossPageIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocBarcodeObjectsCrossPageIteratorImpl * pimpl_

  Pointer to internal implementation.

1.35.1 Detailed Description

Basic const-ref iterator for a collection of barcode objects from several pages.

Definition at line 306 of file doc_physical_document_iterators.h.

1.35.2 Member Data Documentation

pimpl_

```
DocBarcodeObjectsCrossPageIteratorImpl* se::doc::DocBarcodeObjectsCrossPageIterator::pimpl_↩
[private]
```

Pointer to internal implementation.

Definition at line 345 of file doc_physical_document_iterators.h.

1.36 se::doc::DocBarcodeObjectsIterator Class Reference

Public Member Functions

- **DocBarcodeObjectsIterator** (const DocBarcodeObjectsIterator &other)
  Copy ctor.
- DocBarcodeObjectsIterator & **operator=** (const DocBarcodeObjectsIterator &other)
  Assignment operator.
- ~**DocBarcodeObjectsIterator** ()
  Non-trivial dtor.
- const DocBarcodeObject & **GetBarcodeObject** () const
  Returns the barcode object (const ref)
- const DocBarcodeObject * **GetBarcodeObjectPtr** () const
  Returns the barcode object (const ptr)
- void **Advance** ()
  Switches the iterator to point on the next object in its collection.
- bool **Equals** (const DocBarcodeObjectsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocBarcodeObjectsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocBarcodeObjectsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocBarcodeObjectsIterator **ConstructFromImpl** (const DocBarcodeObjectsIteratorImpl &pimpl)
  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocBarcodeObjectsIterator** (const DocBarcodeObjectsIteratorImpl &pimpl)
  Private ctor from internal implementation.

Private Attributes

- DocBarcodeObjectsIteratorImpl * pimpl_
  Pointer to internal implementation.

1.36.1 Detailed Description

Definition at line 230 of file doc_basic_objects_iterator.h.

1.36.2 Member Data Documentation

pimpl_

```
DocBarcodeObjectsIteratorImpl* se::doc::DocBarcodeObjectsIterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 263 of file doc_basic_objects_iterator.h.

1.37 se::doc::DocBaseFieldInfo Class Reference

The class representing basic document field information.

```
#include <doc_fields.h>
```

Public Member Functions

- virtual ~**DocBaseFieldInfo** ()=default
  Default dtor.
- virtual const char * **GetName** () const =0
  Returns the name of the field.
- virtual double **GetConfidence** () const =0
  Returns the confidence of the field (double in range [0.0, 1.0])
- virtual bool **GetAcceptFlag** () const =0
  Returns the field acceptance flag.
- virtual bool **IsValid** () const =0
  Returns the field validity flag.
- virtual bool **IsFictive** () const =0
  Returns true if the field is fictive.
- virtual int **GetAttributesCount** () const =0
  Returns the number of field attributes.
- virtual bool **HasAttribute** (const char *attr_name) const =0
  Returns true iff there exists a field attribute with a given name.
- virtual const char * **GetAttribute** (const char *attr_name) const =0
  Returns the value of an attribute with a given name.
- virtual se::common::StringsMapIterator **AttributesBegin** () const =0
  Returns a 'begin' map-like iterator to the collection of field attributes.
- virtual se::common::StringsMapIterator **AttributesEnd** () const =0
  Returns an 'end' map-like iterator to the collection of field attributes.
- virtual DocTextObjectsCrossPageIterator **ConnectedTextObjectsBegin** (const DocPhysicalDocument &phys_doc) const =0
  Returns a constant 'begin' iterator to all connected text physical objects.
- virtual DocTextObjectsCrossPageIterator **ConnectedTextObjectsEnd** (const DocPhysicalDocument &phys_doc) const =0

Returns a constant 'end' iterator to all connected text physical objects.

- virtual DocTableObjectsCrossPageIterator **ConnectedTableObjectsBegin** (const DocPhysicalDocument &phys_doc) const =0

  Returns a constant 'begin' iterator to all connected table physical objects.
- virtual DocTableObjectsCrossPageIterator **ConnectedTableObjectsEnd** (const DocPhysicalDocument &phys_doc) const =0

  Returns a constant 'end' iterator to all connected table physical objects.
- virtual DocImageObjectsCrossPageIterator **ConnectedImageObjectsBegin** (const DocPhysicalDocument &phys_doc) const =0

  Returns a constant 'begin' iterator to all connected image physical objects.
- virtual DocImageObjectsCrossPageIterator **ConnectedImageObjectsEnd** (const DocPhysicalDocument &phys_doc) const =0

  Returns a constant 'end' iterator to all connected image physical objects.
- virtual DocCheckboxObjectsCrossPageIterator **ConnectedCheckboxObjectsBegin** (const DocPhysicalDocument &phys_doc) const =0

  Returns a constant 'begin' iterator to all connected check box physical objects.
- virtual DocCheckboxObjectsCrossPageIterator **ConnectedCheckboxObjectsEnd** (const DocPhysicalDocument &phys_doc) const =0

  Returns a constant 'end' iterator to all connected check box physical objects.
- virtual DocTextObjectsCrossPageIterator **ConnectedForensicCheckObjectsBegin** (const DocPhysicalDocument &phys_doc) const =0

  Returns a constant 'begin' iterator to all connected check physical objects.
- virtual DocTextObjectsCrossPageIterator **ConnectedForensicCheckObjectsEnd** (const DocPhysicalDocument &phys_doc) const =0

  Returns a constant 'end' iterator to all connected check physical objects.
- virtual DocMetaObjectsCrossPageIterator **ConnectedForensicObjectsBegin** (const DocPhysicalDocument &phys_doc) const =0

  Returns a constant 'begin' iterator to all connected physical objects.
- virtual DocMetaObjectsCrossPageIterator **ConnectedForensicObjectsEnd** (const DocPhysicalDocument &phys_doc) const =0

  Returns a constant 'end' iterator to all connected physical objects.
- virtual DocBarcodeObjectsCrossPageIterator **ConnectedBarcodeObjectsBegin** (const DocPhysicalDocument &phys_doc) const =0

  Returns a constant 'begin' iterator to all connected physical objects.
- virtual DocBarcodeObjectsCrossPageIterator **ConnectedBarcodeObjectsEnd** (const DocPhysicalDocument &phys_doc) const =0

  Returns a constant 'end' iterator to all connected physical objects.
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the field info instance with a given serializer object.
- virtual void SetName (const char *name)=0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.
- virtual void **SetConfidence** (double conf)=0

  Sets the confidence of the field (double in range [0.0, 1.0])
- virtual void **SetAcceptFlag** (bool is_accepted)=0

  Sets the field acceptance flag.
- virtual void **SetAttribute** (const char *attr_name, const char *attr_value)=0

  Sets the field attribute as a key-value pair.
- virtual void **RemoveAttribute** (const char *attr_name)=0

  Removes the field attribute with a given name.
- virtual DocBasicObjectsCrossSliceIterator **ConnectedBasicObjectsBegin** (const DocGraphicalStructure &graphical) const =0

  Returns a constant 'begin' iterator to all connected graphical objects.

- virtual DocBasicObjectsCrossSliceIterator **ConnectedBasicObjectsEnd** (const DocGraphicalStructure &graphical) const =0

  Returns a constant 'end' iterator to all connected graphical objects.
- virtual DocBasicObjectsMutableCrossSliceIterator **MutableConnectedBasicObjectsBegin** (DocGraphicalStructure &graphical)=0

  Returns a mutable 'begin' iterator to all connected graphical objects.
- virtual DocBasicObjectsMutableCrossSliceIterator **MutableConnectedBasicObjectsEnd** (DocGraphicalStructure &graphical)=0

  Returns a mutable 'end' iterator to all connected graphical objects.
- virtual void **ConnectBasicObject** (int coll_id, int obj_id)=0

  Connects a basic object obj_id from collection coll_id with this field.

1.37.1 Detailed Description

The class representing basic document field information.

Definition at line 28 of file doc_fields.h.

1.37.2 Member Function Documentation

SetName()

```
virtual void se::doc::DocBaseFieldInfo::SetName (
            const char * name )  [pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Sets the name of the field

1.38 se::doc::DocBaseObjectInfo Class Reference

The class representing basic information about a graphical object.

```
#include <doc_basic_object.h>
```

Public Member Functions

- virtual ~**DocBaseObjectInfo** ()=default

  Default dtor.
- virtual double **GetConfidence** () const =0

  Returns the object confidence value (double in range [0.0, 1.0])
- virtual bool **GetAcceptFlag** () const =0

  Returns the object acceptance flag.
- virtual const se::common::Polygon & **GetGeometryOnPage** () const =0

  Returns the object geometry in a Polygon form, in a coordinate space of the page image in which this object is placed (const ref)
- virtual const se::common::Polygon * **GetGeometryOnPagePtr** () const =0

Returns the object geometry in a Polygon form, in a coordinate space of the page image in which this object is placed (const ptr)

- virtual const se::common::Polygon & **GetGeometryOnScene** () const =0

  Returns the object geometry in a Polygon form, in a coordinate space of the scene image in which this object is placed (const ref)
- virtual const se::common::Polygon * **GetGeometryOnScenePtr** () const =0

  Returns the object geometry in a Polygon form, in a coordinate space of the scene image in which this object is placed (const ptr)
- virtual int **GetAttributesCount** () const =0

  Gets the number of attributes of the object.
- virtual bool **HasAttribute** (const char *attr_name) const =0

  Returns true iff there is an object attribute with a given name.
- virtual const char * **GetAttribute** (const char *attr_name) const =0

  Returns the value of an object attribute with a given name.
- virtual se::common::StringsMapIterator **AttributesBegin** () const =0

  Return a 'begin' map-like iterator for the object attributes.
- virtual se::common::StringsMapIterator **AttributesEnd** () const =0

  Return an 'end' map-like iterator for the object attributes.
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the object info instance with a given serializer object.
- virtual void SetConfidence (double conf)=0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.
- virtual void **SetAcceptFlag** (bool is_accepted)=0

  Sets the object acceptance flag Method to be deprecated.
- virtual void **SetAttribute** (const char *attr_name, const char *attr_value)=0

  Sets an object attribute (key-value pair) Method to be deprecated.
- virtual void **RemoveAttribute** (const char *attr_name)=0

  Removes the object attribute with a given name Method to be deprecated.
- virtual const se::common::Polygon & **GetGeometry** () const =0

  Returns the object geometry in a Polygon form, in a coordinate space of the collection in which this object is placed (const ref) Method to be deprecated.
- virtual se::common::Polygon & **GetMutableGeometry** ()=0

  Returns the object geometry in a Polygon form, in a coordinate space of the collection in which this object is placed (mutable ref) Method to be deprecated.
- virtual const se::common::Polygon * **GetGeometryPtr** () const =0

  Returns the object geometry in a Polygon form, in a coordinate space of the collection in which this object is placed (const ptr) Method to be deprecated.
- virtual se::common::Polygon * **GetMutableGeometryPtr** ()=0

  Returns the object geometry in a Polygon form, in a coordinate space of the collection in which this object is placed (mutable ptr) Method to be deprecated.
- virtual void **SetGeometry** (const se::common::Polygon &geometry)=0

  Sets the object geometry in a Polygon form, in a coordinate space of the collection in which this object is placed Method to be deprecated.
- virtual int **GetViewID** () const =0

  Returns an ID of a DocView which is associated with this object Method to be deprecated.
- virtual void **SetViewID** (int view_id)=0

  Sets an ID of a DocView which is associated with this object Method to be deprecated.

1.38.1 Detailed Description

The class representing basic information about a graphical object.

Definition at line 23 of file doc_basic_object.h.

1.38.2 Member Function Documentation

SetConfidence()

```
virtual void se::doc::DocBaseObjectInfo::SetConfidence (
            double conf )  [pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

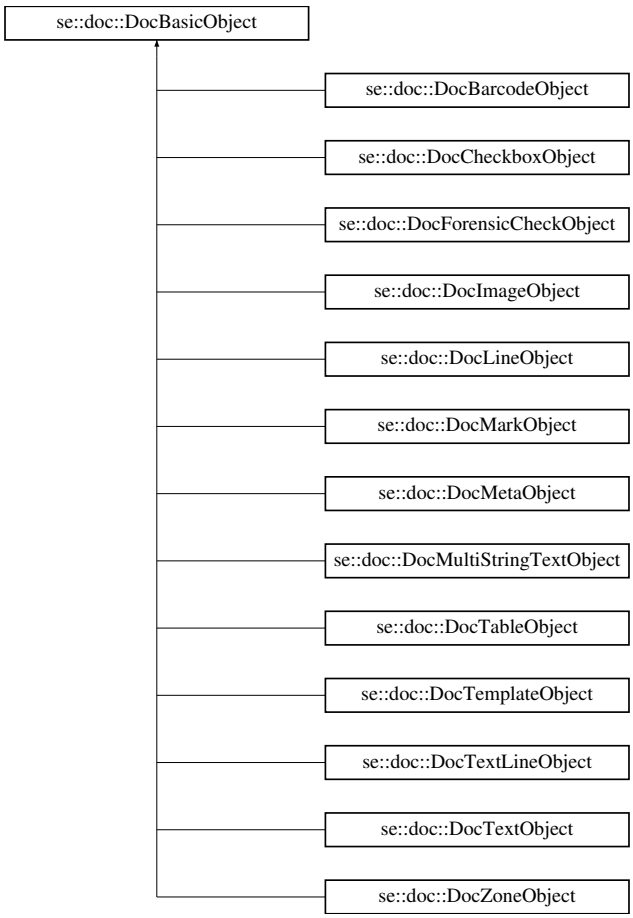
Sets the object confidence value (double in range [0.0, 1.0]) Method to be deprecated

1.39 se::doc::DocBasicObject Class Reference

The class representing a basic graphical object.

```
#include <doc_basic_object.h>
```

Inheritance diagram for se::doc::DocBasicObject:



Public Member Functions

- virtual ~**DocBasicObject** ()=default

  Default dtor.
- virtual const char * **ObjectType** () const =0

  Returns the name of the concrete object type.
- virtual const DocBaseObjectInfo & **GetBaseObjectInfo** () const =0

  Returns the general basic object info (const ref)
- virtual DocBaseObjectInfo & **GetMutableBaseObjectInfo** ()=0

  Returns the general basic object info (mutable ref ref)
- virtual const DocBaseObjectInfo * **GetBaseObjectInfoPtr** () const =0

  Returns the general basic object info (const ptr)
- virtual DocBaseObjectInfo * **GetMutableBaseObjectInfoPtr** ()=0

  Returns the general basic object info (mutable ptr)
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the object instance with a given serializer object.

Static Public Member Functions

- static const char * **BaseClassNameStatic** ()

  Static class name method, returns 'DocBasicObject'.

1.39.1 Detailed Description

The class representing a basic graphical object.

Definition at line 114 of file doc_basic_object.h.

1.40 se::doc::DocBasicObjectsCrossSliceIterator Class Reference

Const-ref iterator for basic objects across multiple collections CLASS TO BE DEPRECATED.

`#include <doc_basic_objects_iterator.h>`

Public Member Functions

- **DocBasicObjectsCrossSliceIterator** (const DocBasicObjectsCrossSliceIterator &other)

  Copy ctor.
- DocBasicObjectsCrossSliceIterator & **operator=** (const DocBasicObjectsCrossSliceIterator &other)

  Assignment operator.
- ~**DocBasicObjectsCrossSliceIterator** ()

  Non-trivial dtor.
- int **GetCollectionID** () const

  Returns the collection ID in which this basic object is placed.
- int **GetObjectID** () const

  Returns the basic object ID.
- const DocBasicObject & **GetBasicObject** () const

  Returns the basic object (const ref)

- const DocTagsCollection & **GetTags** () const

  Returns the tags collection of this object in its collection.
- const DocBasicObject * **GetBasicObjectPtr** () const

  Returns the basic object (const ptr)
- const DocTagsCollection * **GetTagsPtr** () const

  Returns the tags collection of this object in its collection.
- void **Advance** ()

  Switches the iterator to point on the next object.
- bool **Equals** (const DocBasicObjectsCrossSliceIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocBasicObjectsCrossSliceIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocBasicObjectsCrossSliceIterator &rvalue) const

  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocBasicObjectsCrossSliceIterator **ConstructFromImpl** (const DocBasicObjectsCrossSliceIterator↩ Impl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocBasicObjectsCrossSliceIterator** (const DocBasicObjectsCrossSliceIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocBasicObjectsCrossSliceIteratorImpl * pimpl_

  Pointer to internal implementation.

1.40.1 Detailed Description

Const-ref iterator for basic objects across multiple collections CLASS TO BE DEPRECATED.

Definition at line 508 of file doc_basic_objects_iterator.h.

1.40.2 Member Data Documentation

pimpl_

```
DocBasicObjectsCrossSliceIteratorImpl* se::doc::DocBasicObjectsCrossSliceIterator::pimpl_↩
[private]
```

Pointer to internal implementation.

Definition at line 552 of file doc_basic_objects_iterator.h.

1.41 se::doc::DocBasicObjectsIterator Class Reference

Basic const-ref iterator for a collection of basic graphical objects.

`#include <doc_basic_objects_iterator.h>`

Public Member Functions

- **DocBasicObjectsIterator** (const DocBasicObjectsIterator &other)

  Copy ctor.
- DocBasicObjectsIterator & **operator=** (const DocBasicObjectsIterator &other)

  Assignment operator.
- ~**DocBasicObjectsIterator** ()

  Non-trivial dtor.
- int **GetID** () const

  Returns the basic object ID.
- const DocBasicObject & **GetBasicObject** () const

  Returns the basic object (const ref)
- const DocBasicObject * **GetBasicObjectPtr** () const

  Returns the basic object (const ptr)
- void **Advance** ()

  Switches the iterator to point on the next object in its collection.
- bool **Equals** (const DocBasicObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocBasicObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocBasicObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the different objects.
- const DocTagsCollection & GetTags () const

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.
- const DocTagsCollection * **GetTagsPtr** () const

  Returns the tags collection of this object in its collection Method to be deprecated.

Static Public Member Functions

- static DocBasicObjectsIterator **ConstructFromImpl** (const DocBasicObjectsIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocBasicObjectsIterator** (const DocBasicObjectsIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocBasicObjectsIteratorImpl * pimpl_

  Pointer to internal implementation.

1.41.1 Detailed Description

Basic const-ref iterator for a collection of basic graphical objects.

Definition at line 35 of file doc_basic_objects_iterator.h.

1.41.2 Member Function Documentation

GetTags()

```
const DocTagsCollection & se::doc::DocBasicObjectsIterator::GetTags ( ) const
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Returns the tags collection of this object in its collection Method to be deprecated

1.41.3 Member Data Documentation

pimpl_

```
DocBasicObjectsIteratorImpl* se::doc::DocBasicObjectsIterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 82 of file doc_basic_objects_iterator.h.

1.42 se::doc::DocBasicObjectsMutableCrossSliceIterator Class Reference

Mutable-ref iterator for basic objects across multiple collections CLASS TO BE DEPRECATED.

```
#include <doc_basic_objects_iterator.h>
```

Public Member Functions

- **DocBasicObjectsMutableCrossSliceIterator** (const DocBasicObjectsMutableCrossSliceIterator &other)

  Copy ctor.
- DocBasicObjectsMutableCrossSliceIterator & **operator=** (const DocBasicObjectsMutableCrossSliceIterator &other)

  Assignment operator.
- ~**DocBasicObjectsMutableCrossSliceIterator** ()

  Non-trivial dtor.
- int **GetCollectionID** () const

  Returns the collection ID in which this basic object is placed.
- int **GetObjectID** () const

  Returns the basic object ID.
- const DocBasicObject & **GetBasicObject** () const

  Returns the basic object (const ref)

- DocBasicObject & **GetMutableBasicObject** ()

  Returns the basic object (mutable ref)
- const DocTagsCollection & **GetTags** () const

  Returns the tags collection of this object in its collection.
- const DocBasicObject * **GetBasicObjectPtr** () const

  Returns the basic object (const ptr)
- DocBasicObject * **GetMutableBasicObjectPtr** ()

  Returns the basic object (mutable ptr)
- const DocTagsCollection * **GetTagsPtr** () const

  Returns the tags collection of this object in its collection.
- void **Advance** ()

  Switches the iterator to point on the next object.
- bool **Equals** (const DocBasicObjectsMutableCrossSliceIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocBasicObjectsMutableCrossSliceIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocBasicObjectsMutableCrossSliceIterator &rvalue) const

  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocBasicObjectsMutableCrossSliceIterator **ConstructFromImpl** (const DocBasicObjectsMutable↩ CrossSliceIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocBasicObjectsMutableCrossSliceIterator** (const DocBasicObjectsMutableCrossSliceIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocBasicObjectsMutableCrossSliceIteratorImpl * pimpl_

  Pointer to internal implementation.

1.42.1 Detailed Description

Mutable-ref iterator for basic objects across multiple collections CLASS TO BE DEPRECATED.

Definition at line 564 of file doc_basic_objects_iterator.h.

1.42.2 Member Data Documentation

pimpl_

```
DocBasicObjectsMutableCrossSliceIteratorImpl* se::doc::DocBasicObjectsMutableCrossSlice↩
Iterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 614 of file doc_basic_objects_iterator.h.

1.43 se::doc::DocBasicObjectsMutableIterator Class Reference

Mutable-ref iterator for a collection of basic graphical objects CLASS TO BE DEPRECATED.

`#include <doc_basic_objects_iterator.h>`

Public Member Functions

- **DocBasicObjectsMutableIterator** (const DocBasicObjectsMutableIterator &other)

  Copy ctor.
- DocBasicObjectsMutableIterator & **operator=** (const DocBasicObjectsMutableIterator &other)

  Assignment operator.
- ~**DocBasicObjectsMutableIterator** ()

  Non-trivial dtor.
- int **GetID** () const

  Returns the basic object ID.
- const DocBasicObject & **GetBasicObject** () const

  Returns the basic object (const ref)
- DocBasicObject & **GetMutableBasicObject** () const

  Returns the basic object (mutable ref)
- const DocTagsCollection & **GetTags** () const

  Returns the tags collection of this object in its collection.
- const DocBasicObject * **GetBasicObjectPtr** () const

  Returns the basic object (const ptr)
- DocBasicObject * **GetMutableBasicObjectPtr** () const

  Returns the basic object (mutable ptr)
- const DocTagsCollection * **GetTagsPtr** () const

  Returns the tags collection of this object in its collection.
- void **Advance** ()

  Switches the iterator to point on the next object in its collection.
- bool **Equals** (const DocBasicObjectsMutableIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocBasicObjectsMutableIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocBasicObjectsMutableIterator &rvalue) const

  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocBasicObjectsMutableIterator **ConstructFromImpl** (const DocBasicObjectsMutableIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocBasicObjectsMutableIterator** (const DocBasicObjectsMutableIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocBasicObjectsMutableIteratorImpl * pimpl_
  Pointer to internal implementation.

1.43.1 Detailed Description

Mutable-ref iterator for a collection of basic graphical objects CLASS TO BE DEPRECATED.

Definition at line 347 of file doc_basic_objects_iterator.h.

1.43.2 Member Data Documentation

pimpl_

```
DocBasicObjectsMutableIteratorImpl* se::doc::DocBasicObjectsMutableIterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 391 of file doc_basic_objects_iterator.h.

1.44 se::doc::DocBasicObjectsMutableSliceIterator Class Reference

Mutable-ref iterator for a basic objects which have a given tag CLASS TO BE DEPRECATED.

```
#include <doc_basic_objects_iterator.h>
```

Public Member Functions

- **DocBasicObjectsMutableSliceIterator** (const DocBasicObjectsMutableSliceIterator &other)
  Copy ctor.
- DocBasicObjectsMutableSliceIterator & **operator=** (const DocBasicObjectsMutableSliceIterator &other)
  Assignment operator.
- ~**DocBasicObjectsMutableSliceIterator** ()
  Non-trivial dtor.
- int **GetID** () const
  Returns the basic object ID.
- const DocBasicObject & **GetBasicObject** () const
  Returns the basic object (const ref)
- DocBasicObject & **GetMutableBasicObject** () const
  Returns the basic object (mutable ref)
- const DocTagsCollection & **GetTags** () const
  Returns the tags collection of this object in its collection.
- const DocBasicObject * **GetBasicObjectPtr** () const
  Returns the basic object (const ptr)
- DocBasicObject * **GetMutableBasicObjectPtr** () const
  Returns the basic object (mutable ptr)
- const DocTagsCollection * **GetTagsPtr** () const
  Returns the tags collection of this object in its collection.
- void **Advance** ()
  Switches the iterator to point on the next object in its collection.
- bool **Finished** () const
  Returns true iff the iterator points to the end of the subset of objects with a given tag.

Static Public Member Functions

- static DocBasicObjectsMutableSliceIterator **ConstructFromImpl** (const DocBasicObjectsMutableSlice↩ IteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocBasicObjectsMutableSliceIterator** (const DocBasicObjectsMutableSliceIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocBasicObjectsMutableSliceIteratorImpl * pimpl_

  Pointer to internal implementation.

1.44.1 Detailed Description

Mutable-ref iterator for a basic objects which have a given tag CLASS TO BE DEPRECATED.

Definition at line 453 of file doc_basic_objects_iterator.h.

1.44.2 Member Data Documentation

pimpl_

```
DocBasicObjectsMutableSliceIteratorImpl* se::doc::DocBasicObjectsMutableSliceIterator::pimpl_
[private]
```

Pointer to internal implementation.

Definition at line 496 of file doc_basic_objects_iterator.h.

1.45 se::doc::DocBasicObjectsSliceIterator Class Reference

Const-ref iterator for a basic objects which have a given tag CLASS TO BE DEPRECATED.

```
#include <doc_basic_objects_iterator.h>
```

Public Member Functions

- **DocBasicObjectsSliceIterator** (const DocBasicObjectsSliceIterator &other)

  Copy ctor.
- DocBasicObjectsSliceIterator & **operator=** (const DocBasicObjectsSliceIterator &other)

  Assignment operator.
- ~**DocBasicObjectsSliceIterator** ()

  Non-trivial dtor.
- int **GetID** () const

  Returns the basic object ID.
- const DocBasicObject & **GetBasicObject** () const

  Returns the basic object (const ref)
- const DocTagsCollection & **GetTags** () const

  Returns the tags collection of this object in its collection.
- const DocBasicObject * **GetBasicObjectPtr** () const

  Returns the basic object (const ptr)
- const DocTagsCollection * **GetTagsPtr** () const

  Returns the tags collection of this object in its collection.
- void **Advance** ()

  Switches the iterator to point on the next object in its collection.
- bool **Finished** () const

  Returns true iff the iterator points to the end of the subset of objects with a given tag.

Static Public Member Functions

- static DocBasicObjectsSliceIterator **ConstructFromImpl** (const DocBasicObjectsSliceIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocBasicObjectsSliceIterator** (const DocBasicObjectsSliceIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocBasicObjectsSliceIteratorImpl * pimpl_

  Pointer to internal implementation.

1.45.1 Detailed Description

Const-ref iterator for a basic objects which have a given tag CLASS TO BE DEPRECATED.

Definition at line 404 of file doc_basic_objects_iterator.h.

1.45.2 Member Data Documentation

pimpl_

```
DocBasicObjectsSliceIteratorImpl* se::doc::DocBasicObjectsSliceIterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 441 of file doc_basic_objects_iterator.h.

1.46 se::doc::DocCheckboxField Class Reference

The class representing a checkbox field of a document.

```
#include <doc_fields.h>
```

Public Member Functions

- virtual ~**DocCheckboxField** ()=default
  Default dtor.
- virtual const DocBaseFieldInfo & **GetBaseFieldInfo** () const =0
  Returns the basic field information (const ref)
- virtual DocBaseFieldInfo & **GetMutableBaseFieldInfo** ()=0
  Returns the basic field information (mutable ref)
- virtual const DocBaseFieldInfo * **GetBaseFieldInfoPtr** () const =0
  Returns the basic field information (const ptr)
- virtual DocBaseFieldInfo * **GetMutableBaseFieldInfoPtr** ()=0
  Returns the basic field information (mutable ptr)
- virtual bool **GetTickStatus** () const =0
  Returns a boolean ticked status of a checkbox.
- virtual void **SetTickStatus** (bool tick_status)=0
  Sets a ticked status of a checkbox.
- virtual void **Serialize** (se::common::Serializer &serializer) const =0
  Serializes the field instance with a given serializer object.

1.46.1 Detailed Description

The class representing a checkbox field of a document.

Definition at line 216 of file doc_fields.h.

1.47 se::doc::DocCheckboxFieldsIterator Class Reference

Const-ref iterator for a collection of checkbox fields.

```
#include <doc_fields_iterators.h>
```

Public Member Functions

- **DocCheckboxFieldsIterator** (const DocCheckboxFieldsIterator &other)

  Copy ctor.
- DocCheckboxFieldsIterator & **operator=** (const DocCheckboxFieldsIterator &other)

  Assignment operator.
- ~**DocCheckboxFieldsIterator** ()

  Non-trivial dtor.
- const char * **GetKey** () const

  Returns the field name (the collection key)
- const DocCheckboxField & **GetField** () const

  Returns the field value (const ref)
- const DocCheckboxField * **GetFieldPtr** () const

  Returns the field value (const ptr)
- void **Advance** ()

  Switches the iterator to point on the next field in its collection.
- void **operator++** ()

  Switches the iterator to point on the next field in its collection.
- bool **Equals** (const DocCheckboxFieldsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same field.
- bool **operator==** (const DocCheckboxFieldsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same field.
- bool **operator!=** (const DocCheckboxFieldsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the different fields.

Static Public Member Functions

- static DocCheckboxFieldsIterator **ConstructFromImpl** (const DocCheckboxFieldsIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocCheckboxFieldsIterator** (const DocCheckboxFieldsIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- class DocCheckboxFieldsIteratorImpl * pimpl_

  Pointer to internal implementation.

1.47.1 Detailed Description

Const-ref iterator for a collection of checkbox fields.

Definition at line 122 of file doc_fields_iterators.h.

1.47.2 Member Data Documentation

pimpl_

```
class DocCheckboxFieldsIteratorImpl* se::doc::DocCheckboxFieldsIterator::pimpl_  [private]
```

Pointer to internal implementation.

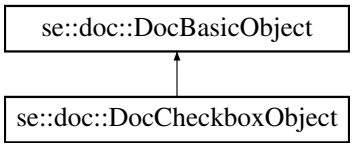
Definition at line 159 of file doc_fields_iterators.h.

1.48 se::doc::DocCheckboxObject Class Reference

The graphical object representing a checkbox.

`#include <doc_objects.h>`

Inheritance diagram for se::doc::DocCheckboxObject:



Public Member Functions

- virtual ~**DocCheckboxObject** () override=default

  Default dtor.
- virtual const se::common::OcrString & **GetOcrString** () const =0

  Returns the OcrString representation of the analysis result (const ref)
- virtual const se::common::OcrString * **GetOcrStringPtr** () const =0

  Returns the OcrString representation of the analysis result (const ptr)
- virtual se::common::OcrString & GetMutableOcrString ()=0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.
- virtual se::common::OcrString * **GetMutableOcrStringPtr** ()=0

  Returns the OcrString representation of the analysis result (mutable ptr)
- virtual void **SetOcrString** (const se::common::OcrString &ocrstring)=0

  Sets the OcrString representation of the analysis result.

Public Member Functions inherited from se::doc::DocBasicObject

- virtual ~**DocBasicObject** ()=default

  Default dtor.
- virtual const char * **ObjectType** () const =0

  Returns the name of the concrete object type.
- virtual const DocBaseObjectInfo & **GetBaseObjectInfo** () const =0

  Returns the general basic object info (const ref)
- virtual DocBaseObjectInfo & **GetMutableBaseObjectInfo** ()=0

  Returns the general basic object info (mutable ref ref)
- virtual const DocBaseObjectInfo * **GetBaseObjectInfoPtr** () const =0

  Returns the general basic object info (const ptr)
- virtual DocBaseObjectInfo * **GetMutableBaseObjectInfoPtr** ()=0

  Returns the general basic object info (mutable ptr)
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the object instance with a given serializer object.

Static Public Member Functions

- static const char * **ObjectTypeStatic** ()
  Returns the object type name.

Static Public Member Functions inherited from se::doc::DocBasicObject

- static const char * **BaseClassNameStatic** ()
  Static class name method, returns 'DocBasicObject'.

1.48.1 Detailed Description

The graphical object representing a checkbox.

Definition at line 116 of file doc_objects.h.

1.48.2 Member Function Documentation

GetMutableOcrString()

```
virtual se::common::OcrString & se::doc::DocCheckboxObject::GetMutableOcrString ( )  [pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Returns the OcrString representation of the analysis result (mutable ref)

1.49 se::doc::DocCheckboxObjectsCrossPageIterator Class Reference

Basic const-ref iterator for a collection of checkbox objects from several pages.

```
#include <doc_physical_document_iterators.h>
```

Public Member Functions

- **DocCheckboxObjectsCrossPageIterator** (const DocCheckboxObjectsCrossPageIterator &other)
  Copy ctor.
- DocCheckboxObjectsCrossPageIterator & **operator=** (const DocCheckboxObjectsCrossPageIterator &other)
  Assignment operator.
- ~**DocCheckboxObjectsCrossPageIterator** ()
  Non-trivial dtor.
- int **GetPhysicalPageID** () const
  Return ID of a physsicak page contaning current object.
- int **GetObjectID** () const
  Return ID of an object.
- const DocCheckboxObject & **GetCheckboxObject** () const
  Returns the checkbox object (const ref)
- const DocCheckboxObject * **GetCheckboxObjectPtr** () const
  Returns the checkbox object (const ptr)
- void **Advance** ()
  Switches the iterator to point on the next object in its collection.
- bool **Equals** (const DocCheckboxObjectsCrossPageIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocCheckboxObjectsCrossPageIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocCheckboxObjectsCrossPageIterator &rvalue) const
  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocCheckboxObjectsCrossPageIterator **ConstructFromImpl** (const DocCheckboxObjectsCross↩ PageIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocCheckboxObjectsCrossPageIterator** (const DocCheckboxObjectsCrossPageIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocCheckboxObjectsCrossPageIteratorImpl * pimpl_

  Pointer to internal implementation.

1.49.1 Detailed Description

Basic const-ref iterator for a collection of checkbox objects from several pages.

Definition at line 217 of file doc_physical_document_iterators.h.

1.49.2 Member Data Documentation

pimpl_

```
DocCheckboxObjectsCrossPageIteratorImpl* se::doc::DocCheckboxObjectsCrossPageIterator::pimpl_
[private]
```

Pointer to internal implementation.

Definition at line 256 of file doc_physical_document_iterators.h.

1.50 se::doc::DocCheckboxObjectsIterator Class Reference

Public Member Functions

- **DocCheckboxObjectsIterator** (const DocCheckboxObjectsIterator &other)

  Copy ctor.
- DocCheckboxObjectsIterator & **operator=** (const DocCheckboxObjectsIterator &other)

  Assignment operator.
- ~**DocCheckboxObjectsIterator** ()

  Non-trivial dtor.
- const DocCheckboxObject & **GetCheckboxObject** () const

  Returns the checkbox object (const ref)
- const DocCheckboxObject * **GetCheckboxObjectPtr** () const

  Returns the checkbox object (const ptr)
- void **Advance** ()

  Switches the iterator to point on the next object in its collection.
- bool **Equals** (const DocCheckboxObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocCheckboxObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocCheckboxObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocCheckboxObjectsIterator **ConstructFromImpl** (const DocCheckboxObjectsIteratorImpl &pimpl)
  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocCheckboxObjectsIterator** (const DocCheckboxObjectsIteratorImpl &pimpl)
  Private ctor from internal implementation.

Private Attributes

- DocCheckboxObjectsIteratorImpl * pimpl_
  Pointer to internal implementation.

1.50.1 Detailed Description

Definition at line 266 of file doc_basic_objects_iterator.h.

1.50.2 Member Data Documentation

pimpl_

```
DocCheckboxObjectsIteratorImpl* se::doc::DocCheckboxObjectsIterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 299 of file doc_basic_objects_iterator.h.

1.51 se::doc::DocDocumentFieldsInfoIterator Class Reference

Const-ref iterator for a collection of document fields info.

```
#include <doc_document_fields_info_iterator.h>
```

Public Member Functions

- **DocDocumentFieldsInfoIterator** (const DocDocumentFieldsInfoIterator &other)

  Copy ctor.
- DocDocumentFieldsInfoIterator & **operator=** (const DocDocumentFieldsInfoIterator &other)

  Assignment operator.
- ~**DocDocumentFieldsInfoIterator** ()

  Non-trivial dtor.
- const char * **GetKey** () const

  Returns the field name (the collection key)
- const DocDocumentFieldInfo & **GetDocumentFieldInfo** () const

  Returns the field value (const ref)
- const DocDocumentFieldInfo * **GetDocumentFieldInfoPtr** () const

  Returns the field value (const ptr)
- void **Advance** ()

  Switches the iterator to point on the next field in its collection.
- void **operator++** ()

  Switches the iterator to point on the next field in its collection.
- bool **Equals** (const DocDocumentFieldsInfoIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same field.
- bool **operator==** (const DocDocumentFieldsInfoIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same field.
- bool **operator!=** (const DocDocumentFieldsInfoIterator &rvalue) const

  Returns true iff this instance and rvalue point to the different fields.

Static Public Member Functions

- static DocDocumentFieldsInfoIterator **ConstructFromImpl** (const DocDocumentFieldsInfoIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocDocumentFieldsInfoIterator** (const DocDocumentFieldsInfoIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- class DocDocumentFieldsInfoIteratorImpl * pimpl_

  Pointer to internal implementation.

1.51.1 Detailed Description

Const-ref iterator for a collection of document fields info.

Definition at line 30 of file doc_document_fields_info_iterator.h.

1.51.2 Member Data Documentation

pimpl_

```
class DocDocumentFieldsInfoIteratorImpl* se::doc::DocDocumentFieldsInfoIterator::pimpl_↩
[private]
```

Pointer to internal implementation.

Definition at line 67 of file doc_document_fields_info_iterator.h.

1.52 se::doc::DocDocumentInfo Class Reference

Reference information about document type.

```
#include <doc_document_info.h>
```

Public Member Functions

- virtual ~**DocDocumentInfo** ()=default

  Default dtor.
- virtual const char * **GetDocumentName** () const =0

  Returns human-readable name of the document.
- virtual const char * **GetDocumentNameLocal** () const =0

  Returns human-readable name of the document in the local language.
- virtual const char * **GetDocumentShortNameLocal** () const =0

  Returns human-readable short name of the document in the local language.
- virtual bool **GetDocumentNoFields** () const =0

  Returns information about whether fields in the document.
- virtual DocDocumentFieldsInfoIterator **DocumentFieldsInfoBegin** () const =0

  Returns a 'begin' iterator to the map of reference information about document fields.
- virtual DocDocumentFieldsInfoIterator **DocumentFieldsInfoEnd** () const =0

  Returns an 'end' iterator to the map of reference information about document fields.
- virtual const DocDocumentFieldInfo & **GetDocumentFieldInfo** (const char *name) const =0

  Returns document field info (const ref)
- virtual const DocDocumentFieldInfo * **GetDocumentFieldInfoPtr** (const char *name) const =0

  Returns document field info (const ptr)
- virtual bool GetDocumentMultipageInfo () const =0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

1.52.1 Detailed Description

Reference information about document type.

Definition at line 22 of file doc_document_info.h.

1.52.2 Member Function Documentation

GetDocumentMultipageInfo()

```
virtual bool se::doc::DocDocumentInfo::GetDocumentMultipageInfo ( ) const  [pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Returns information about whether the document is multipage

1.53 se::doc::DocDocumentTableFieldColumnsInfoInterator Class Reference

Const-ref iterator for a collection of columns inside document table field.

`#include <doc_document_fields_info_iterator.h>`

Public Member Functions

- **DocDocumentTableFieldColumnsInfoInterator** (const DocDocumentTableFieldColumnsInfoInterator &other)

  Copy ctor.
- DocDocumentTableFieldColumnsInfoInterator & **operator=** (const DocDocumentTableFieldColumnsInfoInterator &other)

  Assignment operator.
- ~**DocDocumentTableFieldColumnsInfoInterator** ()

  Non-trivial dtor.
- const char * **GetKey** () const

  Returns the column name (the collection key)
- const DocDocumentTableFieldColumnInfo & **GetDocumentTableFieldColumnInfo** () const

  Returns the column value (const ref)
- const DocDocumentTableFieldColumnInfo * **GetDocumentTableFieldColumnInfoPtr** () const

  Returns the column value (const ptr)
- void **Advance** ()

  Switches the iterator to point on the next column in its collection.
- void **operator++** ()

  Switches the iterator to point on the next column in its collection.
- bool **Equals** (const DocDocumentTableFieldColumnsInfoInterator &rvalue) const

  Returns true iff this instance and rvalue point to the same column.
- bool **operator==** (const DocDocumentTableFieldColumnsInfoInterator &rvalue) const

  Returns true iff this instance and rvalue point to the same column.
- bool **operator!=** (const DocDocumentTableFieldColumnsInfoInterator &rvalue) const

  Returns true iff this instance and rvalue point to the different column.

Static Public Member Functions

- static DocDocumentTableFieldColumnsInfoInterator **ConstructFromImpl** (const DocDocumentTableField↩ ColumnsInfoInteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocDocumentTableFieldColumnsInfoInterator** (const DocDocumentTableFieldColumnsInfoInteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- class DocDocumentTableFieldColumnsInfoInteratorImpl * pimpl_

  Pointer to internal implementation.

1.53.1 Detailed Description

Const-ref iterator for a collection of columns inside document table field.

Definition at line 74 of file doc_document_fields_info_iterator.h.

1.53.2 Member Data Documentation

pimpl_

```
class DocDocumentTableFieldColumnsInfoInteratorImpl* se::doc::DocDocumentTableFieldColumns↩
InfoInterator::pimpl_ [private]
```

Pointer to internal implementation.

Definition at line 112 of file doc_document_fields_info_iterator.h.

1.54 se::doc::DocEngine Class Reference

The main DocEngine class containing all configuration and resources of the Smart Document Engine.

```
#include <doc_engine.h>
```

Public Member Functions

- virtual ~**DocEngine** ()=default

  Default dtor.
- virtual DocSessionSettings * CreateSessionSettings () const =0

  Creates a Session Settings object with default processing and recognition settings, specified in the configuration bundle.
- virtual DocSession * SpawnSession (const DocSessionSettings &settings, const char *signature, DocFeedback *feedback_reporter=nullptr) const =0

  Spawns a new documents recognition session.
- virtual DocSession * SpawnSession (const DocSessionSettings &settings, const char *signature, DocFeedback *feedback_reporter, DocExternalProcessorInterface *external_processor) const =0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.
- virtual DocSessionSettings * CreateVideoSessionSettings () const =0

  DEPRECATED METHODS NO LONGER WORKS THEY ARE TO BE DELETED IN FUTURE VERSIONS.
- virtual DocVideoSession * SpawnVideoSession (const DocSessionSettings &settings, const char *signature, DocFeedback *feedback_reporter=nullptr) const =0

  Spawns a new video stream document recognition session.

Static Public Member Functions

- static DocEngine * Create (const char *config_path, bool lazy_configuration=true)

  The factory method for creating the DocEngine object with a configuration bundle file.
- static DocEngine * Create (unsigned char *config_data, int config_data_length, bool lazy_configuration=true)

  The factory method for creating the DocEngine object with a configuration bundle buffer.
- static DocEngine * CreateFromEmbeddedBundle (bool lazy_configuration=true)

  The factory method for creating the DocEngine object with a configuration bundle buffer embedded within the library.
- static const char * GetVersion ()

  Returns the Smart Document Engine version number.

1.54.1 Detailed Description

The main DocEngine class containing all configuration and resources of the Smart Document Engine.

Definition at line 24 of file doc_engine.h.

1.54.2 Member Function Documentation

CreateSessionSettings()

```
virtual DocSessionSettings * se::doc::DocEngine::CreateSessionSettings ( ) const  [pure virtual]
```

Creates a Session Settings object with default processing and recognition settings, specified in the configuration bundle.

Returns

A newly created DocSessionSettings object. The object is allocated, the caller is responsible for deleting it.

SpawnSession() [1/2]

```
virtual DocSession * se::doc::DocEngine::SpawnSession (
            const DocSessionSettings & settings,
            const char * signature,
            DocFeedback * feedback_reporter = nullptr ) const  [pure virtual]
```

Spawns a new documents recognition session.

Parameters

settings	- a settings object which are used to spawn a session
signature	- a unique caller signature to unlock the internal library calls (provided with your SDK package)
feedback_reporter	- an optional pointer to the implementation of feedback callbacks class

Returns

A newly created session (DocSession object). The object is allocated, the caller is responsible for deleting it.

Create() [1/2]

```
static DocEngine * se::doc::DocEngine::Create (
            const char * config_path,
            bool lazy_configuration = true )  [static]
```

The factory method for creating the DocEngine object with a configuration bundle file.

Parameters

config_path	- filesystem path to an engine configuration bundle
lazy_configuration	- if true, some components of the internal engine structure will be initialized only when first needed. If false, all engine structure will be loaded and initialized immediately. Lazy configuration is enabled by default.

Returns

A newly created DocEngine object. The object is allocated, the caller is responsible for deleting it.

Create() [2/2]

```
static DocEngine * se::doc::DocEngine::Create (
            unsigned char * config_data,
            int config_data_length,
            bool lazy_configuration = true )  [static]
```

The factory method for creating the DocEngine object with a configuration bundle buffer.

Parameters

config_data	- pointer to the configuration bundle file buffer.
config_data_length	- size of the configuration buffer in bytes.
lazy_configuration	- if true, some components of the internal engine structure will be initialized only when first needed. If false, all engine structure will be loaded and initialized immediately. Lazy configuration is enabled by default.

Returns

A newly created DocEngine object. The object is allocated, the caller is responsible for deleting it.

CreateFromEmbeddedBundle()

```
static DocEngine * se::doc::DocEngine::CreateFromEmbeddedBundle (
            bool lazy_configuration = true )  [static]
```

The factory method for creating the DocEngine object with a configuration bundle buffer embedded within the library.

Parameters

lazy_configuration	- if true, some components of the internal engine structure will be initialized only when first needed. If false, all engine structure will be loaded and initialized immediately. Lazy configuration is enabled by default.

Returns

A newly created DocEngine object. The object is allocated, the caller is responsible for deleting it.

GetVersion()

```
static const char * se::doc::DocEngine::GetVersion ( )  [static]
```

Returns the Smart Document Engine version number.

Returns

Smart Document Engine version number string

SpawnSession() [2/2]

```
virtual DocSession * se::doc::DocEngine::SpawnSession (
            const DocSessionSettings & settings,
            const char * signature,
            DocFeedback * feedback_reporter,
            DocExternalProcessorInterface * external_processor ) const  [pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Spawns a new documents recognition session

Parameters

settings	- a settings object which are used to spawn a session
signature	- a unique caller signature to unlock the internal library calls (provided with your SDK package)
feedback_reporter	- an optional pointer to the implementation of feedback callbacks class
external_processor	- an optional pointer to the implementation of an external document processor

Returns

A newly created session (DocSession object). The object is allocated, the caller is responsible for deleting it.

CreateVideoSessionSettings()

```
virtual DocSessionSettings * se::doc::DocEngine::CreateVideoSessionSettings ( ) const  [pure virtual]
```

DEPRECATED METHODS NO LONGER WORKS THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Creates a Video Session Settings object with default processing and recognition settings for a sequence of video frames, specified in the configuration bundle.

Returns

A newly created DocSessionSettings object. The object is allocated, the caller is responsible for deleting it.

SpawnVideoSession()

```
virtual DocVideoSession * se::doc::DocEngine::SpawnVideoSession (
            const DocSessionSettings & settings,
            const char * signature,
            DocFeedback * feedback_reporter = nullptr ) const  [pure virtual]
```

Spawns a new video stream document recognition session.

Parameters

settings	- a settings object which are used to spawn a session
signature	- a unique caller signature to unlock the internal library calls (provided with your SDK package)
feedback_reporter	- an optional pointer to the implementation of the feedback callbacks class

Returns

A newly created video session (DocVideoSession object). The object is allocated, the caller is responsible for deleting it.

1.55 se::doc::DocExternalProcessorInterface Class Reference

The abstract interface for custom document processor CLASS TO BE DEPRECATED.

`#include <doc_external_processor.h>`

Public Member Functions

- virtual ~**DocExternalProcessorInterface** ()=default

  Default dtor.
- virtual void Process (DocResult &recognition_result, const DocProcessingSettings &processing_settings, const DocProcessingArguments &processing_arguments)=0

  Processes the current result structure with a given processing settings and arguments. Needs to be implemented in a derived custom processing class.

1.55.1 Detailed Description

The abstract interface for custom document processor CLASS TO BE DEPRECATED.

Definition at line 46 of file doc_external_processor.h.

1.55.2 Member Function Documentation

Process()

```
virtual void se::doc::DocExternalProcessorInterface::Process (
            DocResult & recognition_result,
            const DocProcessingSettings & processing_settings,
            const DocProcessingArguments & processing_arguments )  [pure virtual]
```

Processes the current result structure with a given processing settings and arguments. Needs to be implemented in a derived custom processing class.

Parameters

recognition_result	- mutable current document processing and recognition result structure.
processing_settings	- current source processing settings
processing_arguments	- processing arguments for the current custom document processor

1.56 se::doc::DocFeedback Class Reference

Abstract interface for receiving Smart Document Engine callbacks. All callbacks must be implemented.

```
#include <doc_feedback.h>
```

Public Member Functions

- virtual ~**DocFeedback** ()=default

  Default dtor.
- virtual void FeedbackReceived (const DocFeedbackContainer &container)=0

  Callback for receiving custom feedback container.
- virtual bool **AcceptsPagesLocalizationFeedback** () const

  Returns true if localization feedback is needed Returns true by default.
- virtual void PagesLocalizationFeedbackReceived (const DocPagesFeedbackContainer &container) const =0

  Callback for receiving feedback container with pages localization results.
- virtual bool **AcceptsPagePreprocessingFeedback** () const

  Returns true if page preprocessing feedback is needed Returns true by default.
- virtual void PagePrepocessingFeedbackReceived (const DocPagesFeedbackContainer &container) const =0

  Callback for receiving feedback container with pages preprocessing results.
- virtual bool **AcceptsRawFieldsLocalizationFeedback** () const

  Returns true if fields' localization feedback is needed Returns true by default.
- virtual void RawFieldsLocalizationFeedbackReceived (const DocRawFieldsFeedbackContainer &container) const =0

  Callback for receiving feedback container with raw fields localization results.
- virtual bool **AcceptsRawFieldsRecognitionFeedback** () const

  Returns true if fields' raw recognition feedback is needed Returns true by default.
- virtual void RawFiedlsRecognitionFeedbackReceived (const DocRawFieldsFeedbackContainer &container) const =0

  Callback for receiving feedback container with raw fields recognition results.
- virtual void ResultReceived (const DocResult &result_received)=0

  Callback for receiving an updated stream recognition result.

1.56.1 Detailed Description

Abstract interface for receiving Smart Document Engine callbacks. All callbacks must be implemented.

Definition at line 128 of file doc_feedback.h.

1.56.2 Member Function Documentation

FeedbackReceived()

```
virtual void se::doc::DocFeedback::FeedbackReceived (
            const DocFeedbackContainer & container )  [pure virtual]
```

Callback for receiving custom feedback container.

Parameters

container	- the received feedback container Method to be deprecated

PagesLocalizationFeedbackReceived()

```
virtual void se::doc::DocFeedback::PagesLocalizationFeedbackReceived (
            const DocPagesFeedbackContainer & container ) const  [pure virtual]
```

Callback for receiving feedback container with pages localization results.

Parameters

container	- the received feedback container

PagePrepocessingFeedbackReceived()

```
virtual void se::doc::DocFeedback::PagePrepocessingFeedbackReceived (
            const DocPagesFeedbackContainer & container ) const  [pure virtual]
```

Callback for receiving feedback container with pages preprocessing results.

Parameters

container	- the received feedback container

RawFieldsLocalizationFeedbackReceived()

```
virtual void se::doc::DocFeedback::RawFieldsLocalizationFeedbackReceived (
            const DocRawFieldsFeedbackContainer & container ) const  [pure virtual]
```

Callback for receiving feedback container with raw fields localization results.

Parameters

container	- the received feedback container

RawFiedlsRecognitionFeedbackReceived()

```
virtual void se::doc::DocFeedback::RawFiedlsRecognitionFeedbackReceived (
            const DocRawFieldsFeedbackContainer & container ) const  [pure virtual]
```

Callback for receiving feedback container with raw fields recognition results.

Parameters

container	- the received feedback container

ResultReceived()

```
virtual void se::doc::DocFeedback::ResultReceived (
            const DocResult & result_received )  [pure virtual]
```

Callback for receiving an updated stream recognition result.

Parameters

result_received	- the received recognition result

1.57 se::doc::DocFeedbackContainer Class Reference

The class representing a custom feedback container. Not implemented in the current version of Smart Document Engine CLASS TO BE DEPRECATED.

`#include <doc_feedback.h>`

Public Member Functions

- virtual ~**DocFeedbackContainer** ()=default

  Default dtor.
- virtual se::common::StringsMapIterator **FeedbackFieldIteratorBegin** () const =0

  Returns a begin-iterator for an internal collection of feedback text fields.
- virtual se::common::StringsMapIterator **FeedbackFieldIteratorEnd** () const =0

  Returns a end-iterator for an internal collection of feedback text fields.
- virtual se::common::QuadranglesMapIterator **FeedbackQuadIteratorBegin** () const =0

  Returns a begin-iterator for an internal collection of feedback quadrangles.
- virtual se::common::QuadranglesMapIterator **FeedbackQuadIteratorEnd** () const =0

  Returns a end-iterator for an internal collection of feedback quadrangles.
- virtual void **SetFeedbackField** (const char *key, const char *field)=0

  Feedback field setter.
- virtual void **SetFeedbackQuad** (const char *key, const se::common::Quadrangle &quad)=0

  Feedback quad setter.

1.57.1 Detailed Description

The class representing a custom feedback container. Not implemented in the current version of Smart Document Engine CLASS TO BE DEPRECATED.

Definition at line 105 of file doc_feedback.h.

1.58 se::doc::DocForensicCheckField Class Reference

The class representing a forensic check field of a document.

`#include <doc_fields.h>`

Public Member Functions

- virtual ~**DocForensicCheckField** ()=default

  Default dtor.
- virtual const DocBaseFieldInfo & **GetBaseFieldInfo** () const =0

  Returns the basic field information (const ref)
- virtual DocBaseFieldInfo & **GetMutableBaseFieldInfo** ()=0

  Returns the basic field information (mutable ref)
- virtual const DocBaseFieldInfo * **GetBaseFieldInfoPtr** () const =0

  Returns the basic field information (const ptr)
- virtual DocBaseFieldInfo * **GetMutableBaseFieldInfoPtr** ()=0

  Returns the basic field information (mutable ptr)
- virtual const char * **GetStatus** () const =0

  Returns a forensic field value.
- virtual void **SetStatus** (const char *status)=0

  Sets a forensic field value.
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the field instance with a given serializer object.

1.58.1 Detailed Description

The class representing a forensic check field of a document.

Definition at line 269 of file doc_fields.h.

1.59 se::doc::DocForensicCheckFieldsIterator Class Reference

Const-ref iterator for a collection of forensic check fields.

`#include <doc_fields_iterators.h>`

Public Member Functions

- **DocForensicCheckFieldsIterator** (const DocForensicCheckFieldsIterator &other)

  Copy ctor.
- DocForensicCheckFieldsIterator & **operator=** (const DocForensicCheckFieldsIterator &other)

  Assignment operator.
- ~**DocForensicCheckFieldsIterator** ()

  Non-trivial dtor.
- const char * **GetKey** () const

  Returns the field name (the collection key)
- const DocForensicCheckField & **GetField** () const

  Returns the field value (const ref)
- const DocForensicCheckField * **GetFieldPtr** () const

  Returns the field value (const ptr)
- void **Advance** ()

  Switches the iterator to point on the next field in its collection.
- void **operator++** ()

  Switches the iterator to point on the next field in its collection.
- bool **Equals** (const DocForensicCheckFieldsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same field.
- bool **operator==** (const DocForensicCheckFieldsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same field.
- bool **operator!=** (const DocForensicCheckFieldsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the different fields.

Static Public Member Functions

- static DocForensicCheckFieldsIterator **ConstructFromImpl** (const DocForensicCheckFieldsIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocForensicCheckFieldsIterator** (const DocForensicCheckFieldsIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- class DocForensicCheckFieldsIteratorImpl * pimpl_

  Pointer to internal implementation.

1.59.1 Detailed Description

Const-ref iterator for a collection of forensic check fields.

Definition at line 217 of file doc_fields_iterators.h.

1.59.2 Member Data Documentation

pimpl_

```
class DocForensicCheckFieldsIteratorImpl* se::doc::DocForensicCheckFieldsIterator::pimpl_↩
[private]
```

Pointer to internal implementation.

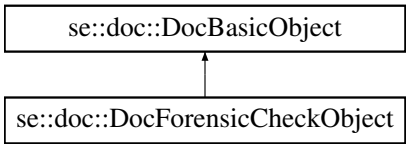
Definition at line 254 of file doc_fields_iterators.h.

1.60 se::doc::DocForensicCheckObject Class Reference

The graphical object representing a forensic check.

```
#include <doc_objects.h>
```

Inheritance diagram for se::doc::DocForensicCheckObject:



Public Member Functions

- virtual ~**DocForensicCheckObject** () override=default

  Default dtor.
- virtual const se::common::OcrString & **GetOcrString** () const =0

  Returns the forensic check result (const ref)
- virtual const se::common::OcrString * **GetOcrStringPtr** () const =0

  Returns the forensic check result (const ptr)
- virtual se::common::OcrString & GetMutableOcrString ()=0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.
- virtual se::common::OcrString * **GetMutableOcrStringPtr** ()=0

  Returns the forensic check result (mutable ptr)

Public Member Functions inherited from se::doc::DocBasicObject

- virtual ~**DocBasicObject** ()=default

  Default dtor.
- virtual const char * **ObjectType** () const =0

  Returns the name of the concrete object type.
- virtual const DocBaseObjectInfo & **GetBaseObjectInfo** () const =0

  Returns the general basic object info (const ref)
- virtual DocBaseObjectInfo & **GetMutableBaseObjectInfo** ()=0

  Returns the general basic object info (mutable ref ref)
- virtual const DocBaseObjectInfo * **GetBaseObjectInfoPtr** () const =0

  Returns the general basic object info (const ptr)
- virtual DocBaseObjectInfo * **GetMutableBaseObjectInfoPtr** ()=0

  Returns the general basic object info (mutable ptr)
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the object instance with a given serializer object.

Static Public Member Functions

- static const char * **ObjectTypeStatic** ()

  Returns the object type name.

Static Public Member Functions inherited from se::doc::DocBasicObject

- static const char * **BaseClassNameStatic** ()

  Static class name method, returns 'DocBasicObject'.

1.60.1 Detailed Description

The graphical object representing a forensic check.

Definition at line 88 of file doc_objects.h.

1.60.2 Member Function Documentation

GetMutableOcrString()

```
virtual se::common::OcrString & se::doc::DocForensicCheckObject::GetMutableOcrString ( )  [pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Returns the forensic check result (mutable ref)

1.61 se::doc::DocForensicCheckObjectsCrossPageIterator Class Reference

Basic const-ref iterator for a collection of forensic check objects from several pages.

```
#include <doc_physical_document_iterators.h>
```

Public Member Functions

- **DocForensicCheckObjectsCrossPageIterator** (const DocForensicCheckObjectsCrossPageIterator &other)

  Copy ctor.
- DocForensicCheckObjectsCrossPageIterator & **operator=** (const DocForensicCheckObjectsCrossPageIterator &other)

  Assignment operator.
- ~**DocForensicCheckObjectsCrossPageIterator** ()

  Non-trivial dtor.
- int **GetPhysicalPageID** () const

  Return ID of a physsicak page contaning current object.
- const DocForensicCheckObject & **GetForensicCheckObject** () const

  Returns the forensic check object (const ref)
- const DocForensicCheckObject * **GetForensicCheckObjectPtr** () const

  Returns the forensic check object (const ptr)
- void **Advance** ()

  Switches the iterator to point on the next object in its collection.
- bool **Equals** (const DocForensicCheckObjectsCrossPageIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocForensicCheckObjectsCrossPageIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocForensicCheckObjectsCrossPageIterator &rvalue) const

  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocForensicCheckObjectsCrossPageIterator **ConstructFromImpl** (const DocForensicCheckObjects↩ CrossPageIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocForensicCheckObjectsCrossPageIterator** (const DocForensicCheckObjectsCrossPageIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocForensicCheckObjectsCrossPageIteratorImpl * pimpl_

  Pointer to internal implementation.

1.61.1 Detailed Description

Basic const-ref iterator for a collection of forensic check objects from several pages.

Definition at line 82 of file doc_physical_document_iterators.h.

1.61.2 Member Data Documentation

pimpl_

```
DocForensicCheckObjectsCrossPageIteratorImpl* se::doc::DocForensicCheckObjectsCrossPage↩
Iterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 118 of file doc_physical_document_iterators.h.

1.62 se::doc::DocForensicCheckObjectsIterator Class Reference

Public Member Functions

- **DocForensicCheckObjectsIterator** (const DocForensicCheckObjectsIterator &other)

  Copy ctor.
- DocForensicCheckObjectsIterator & **operator=** (const DocForensicCheckObjectsIterator &other)

  Assignment operator.
- ~**DocForensicCheckObjectsIterator** ()

  Non-trivial dtor.
- const DocForensicCheckObject & **GetForensicCheckObject** () const

  Returns the forensic check object (const ref)
- const DocForensicCheckObject * **GetForensicCheckObjectPtr** () const

  Returns the forensic check object (const ptr)
- void **Advance** ()

  Switches the iterator to point on the next object in its collection.
- bool **Equals** (const DocForensicCheckObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocForensicCheckObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocForensicCheckObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocForensicCheckObjectsIterator **ConstructFromImpl** (const DocForensicCheckObjectsIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocForensicCheckObjectsIterator** (const DocForensicCheckObjectsIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocForensicCheckObjectsIteratorImpl * pimpl_

  Pointer to internal implementation.

1.62.1 Detailed Description

Definition at line 122 of file doc_basic_objects_iterator.h.

1.62.2 Member Data Documentation

pimpl_

```
DocForensicCheckObjectsIteratorImpl* se::doc::DocForensicCheckObjectsIterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 155 of file doc_basic_objects_iterator.h.

1.63 se::doc::DocForensicField Class Reference

The class representing a forensic field of a document.

```
#include <doc_fields.h>
```

Public Member Functions

- virtual ~**DocForensicField** ()=default

  Default dtor.
- virtual const DocBaseFieldInfo & **GetBaseFieldInfo** () const =0

  Returns the basic field information (const ref)
- virtual DocBaseFieldInfo & **GetMutableBaseFieldInfo** ()=0

  Returns the basic field information (mutable ref)
- virtual const DocBaseFieldInfo * **GetBaseFieldInfoPtr** () const =0

  Returns the basic field information (const ptr)
- virtual DocBaseFieldInfo * **GetMutableBaseFieldInfoPtr** ()=0

  Returns the basic field information (mutable ptr)
- virtual const char * **GetStatus** () const =0

  Returns a forensic field value.
- virtual void **SetStatus** (const char *status)=0

  Sets a forensic field value.
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the field instance with a given serializer object.

1.63.1 Detailed Description

The class representing a forensic field of a document.

Definition at line 243 of file doc_fields.h.

1.64 se::doc::DocForensicFieldsIterator Class Reference

Const-ref iterator for a collection of forensic fields.

`#include <doc_fields_iterators.h>`

Public Member Functions

- **DocForensicFieldsIterator** (const DocForensicFieldsIterator &other)
  Copy ctor.
- DocForensicFieldsIterator & **operator=** (const DocForensicFieldsIterator &other)
  Assignment operator.
- ~**DocForensicFieldsIterator** ()
  Non-trivial dtor.
- const char * **GetKey** () const
  Returns the field name (the collection key)
- const DocForensicField & **GetField** () const
  Returns the field value (const ref)
- const DocForensicField * **GetFieldPtr** () const
  Returns the field value (const ptr)
- void **Advance** ()
  Switches the iterator to point on the next field in its collection.
- void **operator++** ()
  Switches the iterator to point on the next field in its collection.
- bool **Equals** (const DocForensicFieldsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same field.
- bool **operator==** (const DocForensicFieldsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same field.
- bool **operator!=** (const DocForensicFieldsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the different fields.

Static Public Member Functions

- static DocForensicFieldsIterator **ConstructFromImpl** (const DocForensicFieldsIteratorImpl &pimpl)
  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocForensicFieldsIterator** (const DocForensicFieldsIteratorImpl &pimpl)
  Private ctor from internal implementation.

Private Attributes

- class DocForensicFieldsIteratorImpl * pimpl_

  Pointer to internal implementation.

1.64.1 Detailed Description

Const-ref iterator for a collection of forensic fields.

Definition at line 170 of file doc_fields_iterators.h.

1.64.2 Member Data Documentation

pimpl_

```
class DocForensicFieldsIteratorImpl* se::doc::DocForensicFieldsIterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 207 of file doc_fields_iterators.h.

1.65 se::doc::DocGraphicalStructure Class Reference

The class represting a graphical structure - a result of graphical document processing and graphical objects extraction CLASS TO BE DEPRECATED.

```
#include <doc_graphical_structure.h>
```

Public Member Functions

- virtual ~**DocGraphicalStructure** ()=default

  Default dtor.
- virtual int **GetCollectionsCount** () const =0

  Returns the number of object collections.
- virtual bool **HasCollection** (int c_id) const =0

  Returns true iff there is a collection with a given ID.
- virtual const DocObjectsCollection & **GetCollection** (int c_id) const =0

  Returns the collection with a given ID (const ref)
- virtual DocObjectsCollection & **GetMutableCollection** (int c_id)=0

  Returns the collection with a given ID (mutable ref)
- virtual const DocTagsCollection & **GetCollectionTags** (int c_id) const =0

  Returns the tags associated with a collection with a given ID.
- virtual const DocObjectsCollection * **GetCollectionPtr** (int c_id) const =0

  Returns the collection with a given ID (const ptr)
- virtual DocObjectsCollection * **GetMutableCollectionPtr** (int c_id)=0

  Returns the collection with a given ID (mutable ptr)
- virtual const DocTagsCollection * **GetCollectionTagsPtr** (int c_id) const =0

  Returns the tags associated with a collection with a given ID.

- virtual DocObjectsCollectionsMutableIterator **AddCollection** (const DocObjectsCollection &collection)=0

  Adds a new object collection to the graphical structure.
- virtual DocObjectsCollectionsMutableIterator **AddCollection** (const DocObjectsCollection &collection, const DocTagsCollection &tags)=0

  Adds a new object collection to the graphical structure.
- virtual void **SetCollection** (int c_id, const DocObjectsCollection &collection)=0

  Sets a new object collection by the given ID.
- virtual void **RemoveCollection** (int c_id)=0

  Removes the object collection with a given ID.
- virtual DocObjectsCollectionsIterator **ObjectsCollectionsBegin** () const =0

  Returns a constant 'begin' iterator to the object collections.
- virtual DocObjectsCollectionsIterator **ObjectsCollectionsEnd** () const =0

  Returns a constant 'end' iterator to the object collections.
- virtual DocObjectsCollectionsMutableIterator **MutableObjectsCollectionsBegin** ()=0

  Returns a mutable 'begin' iterator to the object collections.
- virtual DocObjectsCollectionsMutableIterator **MutableObjectsCollectionsEnd** ()=0

  Returns a mutable 'end' iterator to the object collections.
- virtual DocObjectsCollectionsSliceIterator **ObjectsCollectionsSlice** (const char *tag) const =0

  Returns a const iterator to the object collections with a given tag.
- virtual DocObjectsCollectionsMutableSliceIterator **MutableObjectsCollectionsSlice** (const char *tag)=0

  Returns a mutable iterator to the object collections with a given tag.
- virtual const DocViewsCollection & **GetViewsCollection** () const =0

  Returns the collection of view in the graphical structure (const ref)
- virtual DocViewsCollection & **GetMutableViewsCollection** ()=0

  Returns the collection of view in the graphical structure (mutable ref)
- virtual const DocViewsCollection * **GetViewsCollectionPtr** () const =0

  Returns the collection of view in the graphical structure (const ptr)
- virtual DocViewsCollection * **GetMutableViewsCollectionPtr** ()=0

  Returns the collection of view in the graphical structure (mutable ptr)
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the instance with a given serializer object.

1.65.1 Detailed Description

The class represting a graphical structure - a result of graphical document processing and graphical objects extraction CLASS TO BE DEPRECATED.

Definition at line 26 of file doc_graphical_structure.h.

1.66 se::doc::DocImageField Class Reference

The class representing an image field of a document.

`#include <doc_fields.h>`

Public Member Functions

- virtual ~**DocImageField** ()=default

  Default dtor.
- virtual const DocBaseFieldInfo & **GetBaseFieldInfo** () const =0

  Returns the basic field information (const ref)
- virtual DocBaseFieldInfo & **GetMutableBaseFieldInfo** ()=0

  Returns the basic field information (mutable ref)
- virtual const DocBaseFieldInfo * **GetBaseFieldInfoPtr** () const =0

  Returns the basic field information (const ptr)
- virtual DocBaseFieldInfo * **GetMutableBaseFieldInfoPtr** ()=0

  Returns the basic field information (mutable ptr)
- virtual const se::common::Image & **GetImage** () const =0

  Returns the image representation of a field (const ref)
- virtual se::common::Image & **GetMutableImage** ()=0

  Returns the image representation of a field (mutable ref)
- virtual const se::common::Image * **GetImagePtr** () const =0

  Returns the image representation of a field (const ptr)
- virtual se::common::Image * **GetMutableImagePtr** ()=0

  Returns the image representation of a field (mutable ptr)
- virtual void **SetImage** (const se::common::Image &image)=0

  Sets the image representation of a field.
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the field instance with a given serializer object.

1.66.1 Detailed Description

The class representing an image field of a document.

Definition at line 183 of file doc_fields.h.

1.67 se::doc::DocImageFieldsIterator Class Reference

Const-ref iterator for a collection of image fields.

`#include <doc_fields_iterators.h>`

Public Member Functions

- **DocImageFieldsIterator** (const DocImageFieldsIterator &other)

  Copy ctor.
- DocImageFieldsIterator & **operator=** (const DocImageFieldsIterator &other)

  Assignment operator.
- ~**DocImageFieldsIterator** ()

  Non-trivial dtor.
- const char * **GetKey** () const

  Returns the field name (the collection key)
- const DocImageField & **GetField** () const

  Returns the field value (const ref)

- const DocImageField * **GetFieldPtr** () const

  Returns the field value (const ptr)
- void **Advance** ()

  Switches the iterator to point on the next field in its collection.
- void **operator++** ()

  Switches the iterator to point on the next field in its collection.
- bool **Equals** (const DocImageFieldsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same field.
- bool **operator==** (const DocImageFieldsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same field.
- bool **operator!=** (const DocImageFieldsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the different fields.

Static Public Member Functions

- static DocImageFieldsIterator **ConstructFromImpl** (const DocImageFieldsIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocImageFieldsIterator** (const DocImageFieldsIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- class DocImageFieldsIteratorImpl * pimpl_

  Pointer to internal implementation.

1.67.1 Detailed Description

Const-ref iterator for a collection of image fields.

Definition at line 74 of file doc_fields_iterators.h.

1.67.2 Member Data Documentation

pimpl_

```
class DocImageFieldsIteratorImpl* se::doc::DocImageFieldsIterator::pimpl_  [private]
```

Pointer to internal implementation.

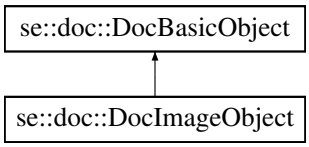
Definition at line 111 of file doc_fields_iterators.h.

1.68 se::doc::DocImageObject Class Reference

The graphical object representing an image region of a document.

`#include <doc_objects.h>`

Inheritance diagram for se::doc::DocImageObject:



Public Member Functions

- virtual ~**DocImageObject** () override=default

  Default dtor.

Public Member Functions inherited from se::doc::DocBasicObject

- virtual ~**DocBasicObject** ()=default

  Default dtor.
- virtual const char * **ObjectType** () const =0

  Returns the name of the concrete object type.
- virtual const DocBaseObjectInfo & **GetBaseObjectInfo** () const =0

  Returns the general basic object info (const ref)
- virtual DocBaseObjectInfo & **GetMutableBaseObjectInfo** ()=0

  Returns the general basic object info (mutable ref ref)
- virtual const DocBaseObjectInfo * **GetBaseObjectInfoPtr** () const =0

  Returns the general basic object info (const ptr)
- virtual DocBaseObjectInfo * **GetMutableBaseObjectInfoPtr** ()=0

  Returns the general basic object info (mutable ptr)
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the object instance with a given serializer object.

Static Public Member Functions

- static const char * **ObjectTypeStatic** ()

  Returns the object type name.

Static Public Member Functions inherited from se::doc::DocBasicObject

- static const char * **BaseClassNameStatic** ()

  Static class name method, returns 'DocBasicObject'.

1.68.1 Detailed Description

The graphical object representing an image region of a document.

Definition at line 311 of file doc_objects.h.

1.69 se::doc::DocImageObjectsCrossPageIterator Class Reference

Basic const-ref iterator for a collection of image objects from several pages.

`#include <doc_physical_document_iterators.h>`

Public Member Functions

- **DocImageObjectsCrossPageIterator** (const DocImageObjectsCrossPageIterator &other)
  Copy ctor.
- DocImageObjectsCrossPageIterator & **operator=** (const DocImageObjectsCrossPageIterator &other)
  Assignment operator.
- ~**DocImageObjectsCrossPageIterator** ()
  Non-trivial dtor.
- int **GetPhysicalPageID** () const
  Return ID of a physical page contaning current object.
- int **GetObjectID** () const
  Return ID of an object.
- const DocImageObject & **GetImageObject** () const
  Returns the image object (const ref)
- const DocImageObject * **GetImageObjectPtr** () const
  Returns the image object (const ptr)
- void **Advance** ()
  Switches the iterator to point on the next object in its collection.
- bool **Equals** (const DocImageObjectsCrossPageIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocImageObjectsCrossPageIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocImageObjectsCrossPageIterator &rvalue) const
  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocImageObjectsCrossPageIterator **ConstructFromImpl** (const DocImageObjectsCrossPage↩IteratorImpl &pimpl)
  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocImageObjectsCrossPageIterator** (const DocImageObjectsCrossPageIteratorImpl &pimpl)
  Private ctor from internal implementation.

Private Attributes

- DocImageObjectsCrossPageIteratorImpl * pimpl_

  Pointer to internal implementation.

1.69.1 Detailed Description

Basic const-ref iterator for a collection of image objects from several pages.

Definition at line 125 of file doc_physical_document_iterators.h.

1.69.2 Member Data Documentation

pimpl_

```
DocImageObjectsCrossPageIteratorImpl* se::doc::DocImageObjectsCrossPageIterator::pimpl_↩
[private]
```

Pointer to internal implementation.

Definition at line 164 of file doc_physical_document_iterators.h.

1.70 se::doc::DocImageObjectsIterator Class Reference

Public Member Functions

- **DocImageObjectsIterator** (const DocImageObjectsIterator &other)

  Copy ctor.
- DocImageObjectsIterator & **operator=** (const DocImageObjectsIterator &other)

  Assignment operator.
- ~**DocImageObjectsIterator** ()

  Non-trivial dtor.
- const DocImageObject & **GetImageObject** () const

  Returns the image object (const ref)
- const DocImageObject * **GetImageObjectPtr** () const

  Returns the image object (const ptr)
- void **Advance** ()

  Switches the iterator to point on the next object in its collection.
- bool **Equals** (const DocImageObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocImageObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocImageObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocImageObjectsIterator **ConstructFromImpl** (const DocImageObjectsIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocImageObjectsIterator** (const DocImageObjectsIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocImageObjectsIteratorImpl * pimpl_

  Pointer to internal implementation.

1.70.1 Detailed Description

Definition at line 158 of file doc_basic_objects_iterator.h.

1.70.2 Member Data Documentation

pimpl_

```
DocImageObjectsIteratorImpl* se::doc::DocImageObjectsIterator::pimpl_  [private]
```

Pointer to internal implementation.

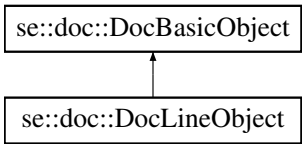
Definition at line 191 of file doc_basic_objects_iterator.h.

1.71 se::doc::DocLineObject Class Reference

The graphical object representing a straight line segment.

```
#include <doc_objects.h>
```

Inheritance diagram for se::doc::DocLineObject:



Public Member Functions

- virtual ~**DocLineObject** () override=default

  Default dtor.

Public Member Functions inherited from se::doc::DocBasicObject

- virtual ~**DocBasicObject** ()=default

  Default dtor.
- virtual const char * **ObjectType** () const =0

  Returns the name of the concrete object type.
- virtual const DocBaseObjectInfo & **GetBaseObjectInfo** () const =0

  Returns the general basic object info (const ref)
- virtual DocBaseObjectInfo & **GetMutableBaseObjectInfo** ()=0

  Returns the general basic object info (mutable ref ref)
- virtual const DocBaseObjectInfo * **GetBaseObjectInfoPtr** () const =0

  Returns the general basic object info (const ptr)
- virtual DocBaseObjectInfo * **GetMutableBaseObjectInfoPtr** ()=0

  Returns the general basic object info (mutable ptr)
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the object instance with a given serializer object.

Static Public Member Functions

- static const char * **ObjectTypeStatic** ()

  Returns the object type name.

Static Public Member Functions inherited from se::doc::DocBasicObject

- static const char * **BaseClassNameStatic** ()

  Static class name method, returns 'DocBasicObject'.

1.71.1 Detailed Description

The graphical object representing a straight line segment.

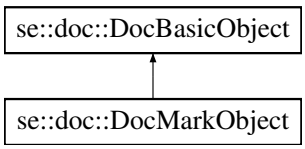
Definition at line 159 of file doc_objects.h.

1.72 se::doc::DocMarkObject Class Reference

The graphical object representing a remark or correction on a document.

```
#include <doc_objects.h>
```

Inheritance diagram for se::doc::DocMarkObject:



Public Member Functions

- virtual ~**DocMarkObject** () override=default
  Default dtor.

Public Member Functions inherited from se::doc::DocBasicObject

- virtual ~**DocBasicObject** ()=default
  Default dtor.
- virtual const char * **ObjectType** () const =0
  Returns the name of the concrete object type.
- virtual const DocBaseObjectInfo & **GetBaseObjectInfo** () const =0
  Returns the general basic object info (const ref)
- virtual DocBaseObjectInfo & **GetMutableBaseObjectInfo** ()=0
  Returns the general basic object info (mutable ref ref)
- virtual const DocBaseObjectInfo * **GetBaseObjectInfoPtr** () const =0
  Returns the general basic object info (const ptr)
- virtual DocBaseObjectInfo * **GetMutableBaseObjectInfoPtr** ()=0
  Returns the general basic object info (mutable ptr)
- virtual void **Serialize** (se::common::Serializer &serializer) const =0
  Serializes the object instance with a given serializer object.

Static Public Member Functions

- static const char * **ObjectTypeStatic** ()
  Returns the object type name.

Static Public Member Functions inherited from se::doc::DocBasicObject

- static const char * **BaseClassNameStatic** ()
  Static class name method, returns 'DocBasicObject'.

1.72.1 Detailed Description

The graphical object representing a remark or correction on a document.

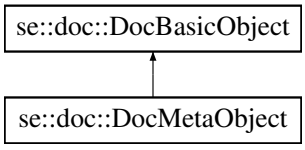
Definition at line 354 of file doc_objects.h.

1.73 se::doc::DocMetaObject Class Reference

The graphical object representing a meta object.

`#include <doc_objects.h>`

Inheritance diagram for se::doc::DocMetaObject:



Public Member Functions

- virtual ~**DocMetaObject** () override=default

  Default dtor.
- virtual const se::common::OcrString & **GetOcrString** () const =0

  Returns the OcrString representation (const ref)
- virtual const se::common::OcrString * **GetOcrStringPtr** () const =0

  Returns the OcrString representation (const ptr)
- virtual se::common::OcrString & GetMutableOcrString ()=0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.
- virtual se::common::OcrString * **GetMutableOcrStringPtr** ()=0

  Returns the OcrString representation (mutable ptr)
- virtual void **SetOcrString** (const se::common::OcrString &ocrstring)=0

  Sets the OcrString representation.

Public Member Functions inherited from se::doc::DocBasicObject

- virtual ~**DocBasicObject** ()=default

  Default dtor.
- virtual const char * **ObjectType** () const =0

  Returns the name of the concrete object type.
- virtual const DocBaseObjectInfo & **GetBaseObjectInfo** () const =0

  Returns the general basic object info (const ref)
- virtual DocBaseObjectInfo & **GetMutableBaseObjectInfo** ()=0

  Returns the general basic object info (mutable ref ref)
- virtual const DocBaseObjectInfo * **GetBaseObjectInfoPtr** () const =0

  Returns the general basic object info (const ptr)
- virtual DocBaseObjectInfo * **GetMutableBaseObjectInfoPtr** ()=0

  Returns the general basic object info (mutable ptr)
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the object instance with a given serializer object.

Static Public Member Functions

- static const char * **ObjectTypeStatic** ()

  Returns the object type name.

Static Public Member Functions inherited from se::doc::DocBasicObject

- static const char * **BaseClassNameStatic** ()

  Static class name method, returns 'DocBasicObject'.

1.73.1 Detailed Description

The graphical object representing a meta object.

Definition at line 228 of file doc_objects.h.

1.73.2 Member Function Documentation

GetMutableOcrString()

```
virtual se::common::OcrString & se::doc::DocMetaObject::GetMutableOcrString ( )  [pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Returns the OcrString representation (mutable ref)

1.74 se::doc::DocMetaObjectsCrossPageIterator Class Reference

Basic const-ref iterator for a collection of meta objects from several pages.

```
#include <doc_physical_document_iterators.h>
```

Public Member Functions

- **DocMetaObjectsCrossPageIterator** (const DocMetaObjectsCrossPageIterator &other)
  Copy ctor.
- DocMetaObjectsCrossPageIterator & **operator=** (const DocMetaObjectsCrossPageIterator &other)
  Assignment operator.
- ~**DocMetaObjectsCrossPageIterator** ()
  Non-trivial dtor.
- int **GetPhysicalPageID** () const
  Return ID of a physsicak page contaning current object.
- const DocMetaObject & **GetMetaObject** () const
  Returns the meta object (const ref)
- const DocMetaObject * **GetMetaObjectPtr** () const
  Returns the meta object (const ptr)
- void **Advance** ()
  Switches the iterator to point on the next object in its collection.
- bool **Equals** (const DocMetaObjectsCrossPageIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocMetaObjectsCrossPageIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocMetaObjectsCrossPageIterator &rvalue) const
  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocMetaObjectsCrossPageIterator **ConstructFromImpl** (const DocMetaObjectsCrossPageIterator↩ Impl &pimpl)
  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocMetaObjectsCrossPageIterator** (const DocMetaObjectsCrossPageIteratorImpl &pimpl)
  Private ctor from internal implementation.

Private Attributes

- DocMetaObjectsCrossPageIteratorImpl * pimpl_

  Pointer to internal implementation.

1.74.1 Detailed Description

Basic const-ref iterator for a collection of meta objects from several pages.

Definition at line 263 of file doc_physical_document_iterators.h.

1.74.2 Member Data Documentation

pimpl_

```
DocMetaObjectsCrossPageIteratorImpl* se::doc::DocMetaObjectsCrossPageIterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 299 of file doc_physical_document_iterators.h.

1.75 se::doc::DocMetaObjectsIterator Class Reference

Public Member Functions

- **DocMetaObjectsIterator** (const DocMetaObjectsIterator &other)

  Copy ctor.
- DocMetaObjectsIterator & **operator=** (const DocMetaObjectsIterator &other)

  Assignment operator.
- ~**DocMetaObjectsIterator** ()

  Non-trivial dtor.
- const DocMetaObject & **GetMetaObject** () const

  Returns the meta object (const ref)
- const DocMetaObject * **GetMetaObjectPtr** () const

  Returns the meta object (const ptr)
- void **Advance** ()

  Switches the iterator to point on the next object in its collection.
- bool **Equals** (const DocMetaObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocMetaObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocMetaObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocMetaObjectsIterator **ConstructFromImpl** (const DocMetaObjectsIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocMetaObjectsIterator** (const DocMetaObjectsIteratorImpl &pimpl)
  Private ctor from internal implementation.

Private Attributes

- DocMetaObjectsIteratorImpl * pimpl_
  Pointer to internal implementation.

1.75.1 Detailed Description

Definition at line 302 of file doc_basic_objects_iterator.h.

1.75.2 Member Data Documentation

pimpl_

```
DocMetaObjectsIteratorImpl* se::doc::DocMetaObjectsIterator::pimpl_  [private]
```

Pointer to internal implementation.

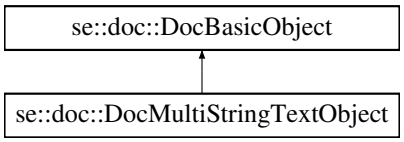
Definition at line 335 of file doc_basic_objects_iterator.h.

1.76 se::doc::DocMultiStringTextObject Class Reference

The graphical object representing a text object with multiple lines CLASS TO BE DEPRECATED.

`#include <doc_objects.h>`

Inheritance diagram for se::doc::DocMultiStringTextObject:



Public Member Functions

- virtual ~**DocMultiStringTextObject** () override=default
  Default dtor.
- virtual int **GetStringsCount** () const =0
  Return the number of text lines.
- virtual void **SetStringsCount** (int count)=0
  Sets the number of text lines.
- virtual const DocTextObject & **GetStringObject** (int index) const =0
  Returns the text object by line index (const ref)
- virtual DocTextObject & **GetMutableStringObject** (int index)=0
  Returns the text object by line index (mutable ref)
- virtual const DocTextObject * **GetStringObjectPtr** (int index) const =0
  Returns the text object by line index (const ptr)
- virtual DocTextObject * **GetMutableStringObjectPtr** (int index)=0
  Returns the text object by line index (mutable ptr)
- virtual void **SetStringObject** (int index, const DocTextObject &text_object)=0
  Sets the text object by line index.

Public Member Functions inherited from se::doc::DocBasicObject

- virtual ~**DocBasicObject** ()=default

  Default dtor.
- virtual const char * **ObjectType** () const =0

  Returns the name of the concrete object type.
- virtual const DocBaseObjectInfo & **GetBaseObjectInfo** () const =0

  Returns the general basic object info (const ref)
- virtual DocBaseObjectInfo & **GetMutableBaseObjectInfo** ()=0

  Returns the general basic object info (mutable ref ref)
- virtual const DocBaseObjectInfo * **GetBaseObjectInfoPtr** () const =0

  Returns the general basic object info (const ptr)
- virtual DocBaseObjectInfo * **GetMutableBaseObjectInfoPtr** ()=0

  Returns the general basic object info (mutable ptr)
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the object instance with a given serializer object.

Static Public Member Functions

- static const char * **ObjectTypeStatic** ()

  Returns the object type name.

Static Public Member Functions inherited from se::doc::DocBasicObject

- static const char * **BaseClassNameStatic** ()

  Static class name method, returns 'DocBasicObject'.

1.76.1 Detailed Description

The graphical object representing a text object with multiple lines CLASS TO BE DEPRECATED.

Definition at line 198 of file doc_objects.h.

1.77 se::doc::DocObjectsCollection Class Reference

The class representing a collection of graphical objects CLASS TO BE DEPRECATED.

`#include <doc_objects_collection.h>`

Public Member Functions

- virtual DocBasicObject * CreateObject () const =0

  Factory method, creates a new graphical object of the type stored in this object collection instance.
- virtual ~**DocObjectsCollection** ()=default

  Default dtor.
- virtual DocObjectsCollection * Clone () const =0

  Clones the collection, returning a deep copy.
- virtual const char * **ObjectType** () const =0

  Returns the name of the stored graphical object type.
- virtual int **GetFrameID** () const =0

  Returns the view ID on which the collected objects are placed.
- virtual void **SetFrameID** (int frame_id)=0

  Sets the view ID on which the collected objects are placed.
- virtual int **GetObjectsCount** () const =0

  Returns the number of stored objects.
- virtual bool **HasObject** (int obj_id) const =0

  Returns true iff there is a stored object with a given ID.
- virtual const DocBasicObject & **GetObject** (int obj_id) const =0

  Returns the object with a given ID (const ref)
- virtual DocBasicObject & **GetMutableObject** (int obj_id)=0

  Returns the object with a given ID (mutable ref)
- virtual const DocBasicObject * **GetObjectPtr** (int obj_id) const =0

  Returns the object with a given ID (const ptr)
- virtual DocBasicObject * **GetMutableObjectPtr** (int obj_id)=0

  Returns the object with a given ID (mutable ptr)
- virtual const DocTagsCollection & **GetObjectTags** (int obj_id) const =0

  Returns the tags collection associated with an object with a given ID.
- virtual const DocTagsCollection * **GetObjectTagsPtr** (int obj_id) const =0

  Returns the tags collection associated with an object with a given ID.
- virtual DocBasicObjectsMutableIterator **AddObject** (const DocBasicObject &obj)=0

  Adds a new object to the collection.
- virtual void **SetObject** (int obj_id, const DocBasicObject &obj)=0

  Sets an object value with a given ID.
- virtual void **RemoveObject** (int obj_id)=0

  Removes an object with a given ID.
- virtual void **RemoveObjectDeep** (int obj_id, DocViewsCollection &views_collection)=0

  Removes an object with a given ID and removes the view associated with the removed object.
- virtual DocBasicObjectsIterator **BasicObjectsBegin** () const =0

  Returns a constant 'begin' iterator to the collection of objects.
- virtual DocBasicObjectsIterator **BasicObjectsEnd** () const =0

  Returns a constant 'end' iterator to the collection of objects.
- virtual DocBasicObjectsMutableIterator **MutableBasicObjectsBegin** ()=0

  Returns a mutable 'begin' iterator to the collection of objects.
- virtual DocBasicObjectsMutableIterator **MutableBasicObjectsEnd** ()=0

  Returns a mutable 'end' iterator to the collection of objects.
- virtual DocBasicObjectsSliceIterator **BasicObjectsSlice** (const char *tag) const =0

  Returns a constant iterator to the subset of objects with a given tag.
- virtual DocBasicObjectsMutableSliceIterator **MutableBasicObjectsSlice** (const char *tag)=0

  Returns a mutable iterator to the subset of objects with a given tag.
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the collection instance with a given serializer object.

Static Public Member Functions

- static const char * **BaseClassNameStatic** ()

  Service method, returns the object name.
- static DocObjectsCollection * Create (const char *object_type)

  Factory method, creates a new collection for objects with a given object type name.

1.77.1 Detailed Description

The class representing a collection of graphical objects CLASS TO BE DEPRECATED.

Definition at line 28 of file doc_objects_collection.h.

1.77.2 Member Function Documentation

Create()

```
static DocObjectsCollection * se::doc::DocObjectsCollection::Create (
            const char * object_type )  [static]
```

Factory method, creates a new collection for objects with a given object type name.

Parameters

object_type	- the name of the graphical object type

Returns

A newly created collection. The object is allocated, the caller is responsible for deleting it.

CreateObject()

```
virtual DocBasicObject * se::doc::DocObjectsCollection::CreateObject ( ) const  [pure virtual]
```

Factory method, creates a new graphical object of the type stored in this object collection instance.

Returns

A newly created graphical object. The object is allocated, the caller is responsible for deleting it.

Clone()

```
virtual DocObjectsCollection * se::doc::DocObjectsCollection::Clone ( ) const  [pure virtual]
```

Clones the collection, returning a deep copy.

Returns

A newly created collection. The object is allocated, the caller is responsible for deleting it.

1.78 se::doc::DocObjectsCollectionsIterator Class Reference

Basic const-ref iterator for graphical object collections CLASS TO BE DEPRECATED.

`#include <doc_objects_collections_iterator.h>`

Public Member Functions

- **DocObjectsCollectionsIterator** (const DocObjectsCollectionsIterator &other)
  Copy ctor.
- DocObjectsCollectionsIterator & **operator=** (const DocObjectsCollectionsIterator &other)
  Assignment operator.
- ~**DocObjectsCollectionsIterator** ()
  Non-trivial dtor.
- int **GetID** () const
  Returns the collection ID.
- const DocObjectsCollection & **GetObjectsCollection** () const
  Returns the collection (const ref)
- const DocTagsCollection & **GetTags** () const
  Returns the tags collection associated with this collection.
- const DocObjectsCollection * **GetObjectsCollectionPtr** () const
  Returns the collection (const ptr)
- const DocTagsCollection * **GetTagsPtr** () const
  Returns the tags collection associated with this collection.
- void **Advance** ()
  Switches the iterator to point on the next collection.
- bool **Equals** (const DocObjectsCollectionsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same collection.
- bool **operator==** (const DocObjectsCollectionsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same collection.
- bool **operator!=** (const DocObjectsCollectionsIterator &rvalue) const
  Returns true iff this instance and rvalue point to a different collection.

Static Public Member Functions

- static DocObjectsCollectionsIterator **ConstructFromImpl** (const DocObjectsCollectionsIteratorImpl &pimpl)
  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocObjectsCollectionsIterator** (const DocObjectsCollectionsIteratorImpl &pimpl)
  Private ctor from internal implementation.

Private Attributes

- DocObjectsCollectionsIteratorImpl * pimpl_
  Pointer to internal implementation.

1.78.1 Detailed Description

Basic const-ref iterator for graphical object collections CLASS TO BE DEPRECATED.

Definition at line 27 of file doc_objects_collections_iterator.h.

1.78.2 Member Data Documentation

pimpl_

```
DocObjectsCollectionsIteratorImpl* se::doc::DocObjectsCollectionsIterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 68 of file doc_objects_collections_iterator.h.

1.79 se::doc::DocObjectsCollectionsMutableIterator Class Reference

Mutable-ref iterator for graphical object collections.

```
#include <doc_objects_collections_iterator.h>
```

Public Member Functions

- **DocObjectsCollectionsMutableIterator** (const DocObjectsCollectionsMutableIterator &other)
  Copy ctor.
- DocObjectsCollectionsMutableIterator & **operator=** (const DocObjectsCollectionsMutableIterator &other)
  Assignment operator.
- ~**DocObjectsCollectionsMutableIterator** ()
  Non-trivial dtor.
- int **GetID** () const
  Returns the collection ID.
- const DocObjectsCollection & **GetObjectsCollection** () const
  Returns the collection (const ptr)
- DocObjectsCollection & **GetMutableObjectsCollection** () const
  Returns the collection (mutable ptr)
- const DocTagsCollection & **GetTags** () const
  Returns the tags collection associated with this collection.
- const DocObjectsCollection * **GetObjectsCollectionPtr** () const
  Returns the collection (const ptr)
- DocObjectsCollection * **GetMutableObjectsCollectionPtr** () const
  Returns the collection (mutable ptr)
- const DocTagsCollection * **GetTagsPtr** () const
  Returns the tags collection associated with this collection.
- void **Advance** ()
  Switches the iterator to point on the next collection.
- bool **Equals** (const DocObjectsCollectionsMutableIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same collection.
- bool **operator==** (const DocObjectsCollectionsMutableIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same collection.
- bool **operator!=** (const DocObjectsCollectionsMutableIterator &rvalue) const
  Returns true iff this instance and rvalue point to a different collection.

Static Public Member Functions

- static DocObjectsCollectionsMutableIterator **ConstructFromImpl** (const DocObjectsCollectionsMutable↩ IteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocObjectsCollectionsMutableIterator** (const DocObjectsCollectionsMutableIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocObjectsCollectionsMutableIteratorImpl * pimpl_

  Pointer to internal implementation.

1.79.1 Detailed Description

Mutable-ref iterator for graphical object collections.

Definition at line 79 of file doc_objects_collections_iterator.h.

1.79.2 Member Data Documentation

pimpl_

```
DocObjectsCollectionsMutableIteratorImpl* se::doc::DocObjectsCollectionsMutableIterator↩
::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 126 of file doc_objects_collections_iterator.h.

1.80 se::doc::DocObjectsCollectionsMutableSliceIterator Class Reference

Const-ref iterator for object collections with a given tag.

```
#include <doc_objects_collections_iterator.h>
```

Public Member Functions

- **DocObjectsCollectionsMutableSliceIterator** (const DocObjectsCollectionsMutableSliceIterator &other)

  Copy ctor.
- DocObjectsCollectionsMutableSliceIterator & **operator=** (const DocObjectsCollectionsMutableSliceIterator &other)

  Assignment operator.
- ~**DocObjectsCollectionsMutableSliceIterator** ()

  Non-trivial dtor.
- int **GetID** () const

  Returns the collection ID.
- const DocObjectsCollection & **GetObjectsCollection** () const

  Returns the collection by const-ref.
- DocObjectsCollection & **GetMutableObjectsCollection** () const

  Returns the collection by mutable-ref.
- const DocTagsCollection & **GetTags** () const

  Returns the tags collection associated with this collection.
- const DocObjectsCollection * **GetObjectsCollectionPtr** () const

  Returns the collection (const ptr)
- DocObjectsCollection * **GetMutableObjectsCollectionPtr** () const

  Returns the collection (mutable ptr)
- const DocTagsCollection * **GetTagsPtr** () const

  Returns the tags collection associated with this collection.
- void **Advance** ()

  Switches the iterator to point on the next collection.
- bool **Finished** () const

  Returns true iff the iterator points to the end of the subset of collections with a given tag.

Static Public Member Functions

- static DocObjectsCollectionsMutableSliceIterator **ConstructFromImpl** (const DocObjectsCollections↩ MutableSliceIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocObjectsCollectionsMutableSliceIterator** (const DocObjectsCollectionsMutableSliceIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocObjectsCollectionsMutableSliceIteratorImpl * pimpl_

  Pointer to internal implementation.

1.80.1 Detailed Description

Const-ref iterator for object collections with a given tag.

Definition at line 189 of file doc_objects_collections_iterator.h.

1.80.2 Member Data Documentation

pimpl_

```
DocObjectsCollectionsMutableSliceIteratorImpl* se::doc::DocObjectsCollectionsMutableSlice↩
Iterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 232 of file doc_objects_collections_iterator.h.

1.81 se::doc::DocObjectsCollectionsSliceIterator Class Reference

Const-ref iterator for graphical object collections with a given tag.

```
#include <doc_objects_collections_iterator.h>
```

Public Member Functions

- **DocObjectsCollectionsSliceIterator** (const DocObjectsCollectionsSliceIterator &other)
  Copy ctor.
- DocObjectsCollectionsSliceIterator & **operator=** (const DocObjectsCollectionsSliceIterator &other)
  Assignment operator.
- ~**DocObjectsCollectionsSliceIterator** ()
  Non-trivial dtor.
- int **GetID** () const
  Returns the collection ID.
- const DocObjectsCollection & **GetObjectsCollection** () const
  Returns the collection by const-ref.
- const DocTagsCollection & **GetTags** () const
  Returns the tags collection associated with this collection.
- const DocObjectsCollection * **GetObjectsCollectionPtr** () const
  Returns the collection (const ptr)
- const DocTagsCollection * **GetTagsPtr** () const
  Returns the tags collection associated with this collection.
- void **Advance** ()
  Switches the iterator to point on the next collection.
- bool **Finished** () const
  Returns true iff the iterator points to the end of the subset of collections with a given tag.

Static Public Member Functions

- static DocObjectsCollectionsSliceIterator **ConstructFromImpl** (const DocObjectsCollectionsSliceIterator↩Impl &pimpl)
  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocObjectsCollectionsSliceIterator** (const DocObjectsCollectionsSliceIteratorImpl &pimpl)
  Private ctor from internal implementation.

Private Attributes

- DocObjectsCollectionsSliceIteratorImpl * pimpl_

  Pointer to internal implementation.

1.81.1 Detailed Description

Const-ref iterator for graphical object collections with a given tag.

Definition at line 138 of file doc_objects_collections_iterator.h.

1.81.2 Member Data Documentation

pimpl_

```
DocObjectsCollectionsSliceIteratorImpl* se::doc::DocObjectsCollectionsSliceIterator::pimpl_↩
[private]
```

Pointer to internal implementation.

Definition at line 177 of file doc_objects_collections_iterator.h.

1.82 se::doc::DocPageFeedback Class Reference

The class representing a feedback for one page.

```
#include <doc_feedback.h>
```

Public Member Functions

- virtual ~**DocPageFeedback** ()=default

  Default dtor.
- virtual const se::common::Quadrangle & **GetQuadrangle** () const =0

  Returns page quadrangle in the original scene.
- virtual int **GetID** () const =0

  Return ID of the page.
- virtual const char * **GetType** () const =0

  Returns document type of teh page.
- virtual bool **IsPageRejected** () const =0

  Return 'true' if the page is not to be processed.

1.82.1 Detailed Description

The class representing a feedback for one page.

Definition at line 67 of file doc_feedback.h.

1.83 se::doc::DocPageInfo Class Reference

The additional infomation about a processed physical page.

`#include <doc_physical_document.h>`

Public Member Functions

- virtual ~**DocPageInfo** ()=default

  Default dtor.
- virtual bool **IsGarbage** () const =0

  Returns true if the page is invalid.
- virtual int **GarbageReasonsCount** () const =0

  Returns the length of the array with reasons why the page is not valid.
- virtual const char * **GarbageReason** (int idx) const =0

  Returns the reason with index idx from the array.

1.83.1 Detailed Description

The additional infomation about a processed physical page.

Definition at line 27 of file doc_physical_document.h.

1.84 se::doc::DocPagesFeedbackContainer Class Reference

The class representing a feedback container for pages.

`#include <doc_feedback.h>`

Public Member Functions

- virtual ~**DocPagesFeedbackContainer** ()=default

  Default dtor.
- virtual int **GetPageCount** () const =0

  Returns the number of pages.
- virtual const DocPageFeedback & **GetPageFeedback** (const int idx) const =0

  Return feedback for the page with given indice.

1.84.1 Detailed Description

The class representing a feedback container for pages.

Definition at line 88 of file doc_feedback.h.

1.85 se::doc::DocPhysicalDocument Class Reference

The class representing the found physical document.

`#include <doc_physical_document.h>`

Public Member Functions

- virtual ~**DocPhysicalDocument** ()=default

  Default dtor.
- virtual int **GetTextObjectsCount** (const char *name) const =0

  Returns a number of text objects connected with a field with a given name.
- virtual int **GetTableObjectsCount** (const char *name) const =0

  Returns a number of table objects connected with a field with a given name.
- virtual int **GetImageObjectsCount** (const char *name) const =0

  Returns a number of image objects connected with a field with a given name.
- virtual int **GetForensicObjectsCount** (const char *name) const =0

  Returns a number of forensic objects connected with a field with a given name.
- virtual int **GetForensicCheckObjectsCount** (const char *name) const =0

  Returns a number of forensic check objects connected with a field with a given name.
- virtual int **GetBarcodeObjectsCount** (const char *name) const =0

  Returns a number of barcode objects connected with a field with a given name.
- virtual int **GetCheckboxObjectsCount** (const char *name) const =0

  Returns a number of checkbox objects connected with a field with a given name.
- virtual int **GetPagesCount** () const =0

  Returns a number of pages in the document.
- virtual const DocPhysicalPage & **GetPhysicalPage** (int idx) const =0

  Returns a page by indices (const ref)
- virtual const DocPhysicalPage * **GetPhysicalPagePtr** (int idx) const =0

  Returns a page by indices (const ptr)
- virtual int GetBasicObjectsCount (const char *name) const =0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

1.85.1 Detailed Description

The class representing the found physical document.

Definition at line 155 of file doc_physical_document.h.

1.85.2 Member Function Documentation

GetBasicObjectsCount()

```
virtual int se::doc::DocPhysicalDocument::GetBasicObjectsCount (
            const char * name ) const  [pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Returns a number of objects connected with a field with a given name

1.86 se::doc::DocPhysicalPage Class Reference

The class representing the found physical page.

`#include <doc_physical_document.h>`

Public Member Functions

- virtual ~**DocPhysicalPage** ()=default

  Default dtor.
- virtual int **GetSourceSceneID** () const =0

  Return an index of the scene.
- virtual DocTextObjectsIterator **TextObjectsBegin** (const char *name) const =0

  Returns a constant 'begin' iterator to the collection of text objects connected with a field with a given name.
- virtual DocTextObjectsIterator **TextObjectsEnd** (const char *name) const =0

  Returns a constant 'end' iterator to the collection of text objects connected with a field with a given name.
- virtual DocImageObjectsIterator **ImageObjectsBegin** (const char *name) const =0

  Returns a constant 'begin' iterator to the collection of image objects connected with a field with a given name.
- virtual DocImageObjectsIterator **ImageObjectsEnd** (const char *name) const =0

  Returns a constant 'end' iterator to the collection of image objects connected with a field with a given name.
- virtual DocTableObjectsIterator **TableObjectsBegin** (const char *name) const =0

  Returns a constant 'begin' iterator to the collection of table objects connected with a field with a given name.
- virtual DocTableObjectsIterator **TableObjectsEnd** (const char *name) const =0

  Returns a constant 'end' iterator to the collection of table objects connected with a field with a given name.
- virtual DocBarcodeObjectsIterator **BarcodeObjectsBegin** (const char *name) const =0

  Returns a constant 'begin' iterator to the collection of barcode objects connected with a field with a given name.
- virtual DocBarcodeObjectsIterator **BarcodeObjectsEnd** (const char *name) const =0

  Returns a constant 'end' iterator to the collection of barcode objects connected with a field with a given name.
- virtual DocCheckboxObjectsIterator **CheckboxObjectsBegin** (const char *name) const =0

  Returns a constant 'begin' iterator to the collection of checkbox objects connected with a field with a given name.
- virtual DocCheckboxObjectsIterator **CheckboxObjectsEnd** (const char *name) const =0

  Returns a constant 'end' iterator to the collection of checkbox objects connected with a field with a given name.
- virtual DocMetaObjectsIterator **ForensicObjectsBegin** (const char *name) const =0

  Returns a constant 'begin' iterator to the collection of forensic objects connected with a field with a given name.
- virtual DocMetaObjectsIterator **ForensicObjectsEnd** (const char *name) const =0

  Returns a constant 'end' iterator to the collection of forensic objects connected with a field with a given name.
- virtual DocForensicCheckObjectsIterator **ForensicCheckObjectsBegin** (const char *name) const =0

  Returns a constant 'begin' iterator to the collection of forensic check objects connected with a field with a given name.
- virtual DocForensicCheckObjectsIterator **ForensicCheckObjectsEnd** (const char *name) const =0

  Returns a constant 'end' iterator to the collection of forensic check objects connected with a field with a given name.
- virtual int **GetTextObjectsCount** (const char *name) const =0

  Returns a number of text objects connected with a field with a given name.
- virtual int **GetImageObjectsCount** (const char *name) const =0

  Returns a number of image objects connected with a field with a given name.
- virtual int **GetTableObjectsCount** (const char *name) const =0

  Returns a number of table objects connected with a field with a given name.
- virtual int **GetBarcodeObjectsCount** (const char *name) const =0

  Returns a number of barcode objects connected with a field with a given name.
- virtual int **GetCheckboxObjectsCount** (const char *name) const =0

  Returns a number of checkbox objects connected with a field with a given name.

- virtual int **GetForensicObjectsCount** (const char *name) const =0

  Returns a number of forensic objects connected with a field with a given name.
- virtual int **GetForensicCheckObjectsCount** (const char *name) const =0

  Returns a number of forensic check objects connected with a field with a given name.
- virtual bool **HasBasicObjects** () const =0

  Returns true if page has at least one object.
- virtual const DocPageInfo & **GetPageInfo** () const =0

  Returns page information, in particular reasons of why page is not valid (const ref)
- virtual const DocPageInfo * **GetPageInfoPtr** () const =0

  Returns page information, in particular reasons of why page is not valid (const ptr)
- virtual const se::common::Quadrangle & **GetPageQuadrangle** () const =0

  Returns the quadrangle of the page in the original image (const ref)
- virtual const se::common::Polygon & **GetPagePolygon** () const =0

  Returns the polygon of the page in the original image.
- virtual const se::common::Quadrangle * **GetPageQuadranglePtr** () const =0

  Returns the quadrangle of the page in the original image (const ptr)
- virtual const se::common::Polygon * **GetPagePolygonPtr** () const =0

  Returns the polygon of the page in the original image (const ptr)
- virtual DocTextObjectsIterator **GetFulltextBasicObjectsBegin** () const =0

  Returns a constant 'begin' iterator to the collection of text objects.
- virtual DocTextObjectsIterator **GetFulltextBasicObjectsEnd** () const =0

  Returns a constant 'end' iterator to the collection of text objects.
- virtual se::common::Image * **GetPageImageFromScene** (const se::common::Image &scene_image) const =0

  Returns image of the page.
- virtual DocTextObjectsIterator **RawTextObjectsBegin** () const =0

  Returns a constant 'begin' iterator to the collection of raw text objects.
- virtual DocTextObjectsIterator **RawTextObjectsEnd** () const =0

  Returns a constant 'end' iterator to the collection of raw text objects.
- virtual int **GetRawTextObjectsCount** () const =0

  Returns a number of raw text objects.
- virtual bool **HasRawTextObject** (const char *name) const =0

  Returns true if page has a raw text object by name.
- virtual const se::doc::DocTextObject & **GetRawTextObject** (const char *name) const =0

  Returns a raw text object by name.
- virtual DocBasicObjectsIterator BasicObjectsBegin (const char *name) const =0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.
- virtual DocBasicObjectsIterator **BasicObjectsEnd** (const char *name) const =0

  Returns a constant 'end' iterator to the collection of basic objects connected with a field with a given name.
- virtual int **GetBasicObjectsCount** (const char *name) const =0

  Returns a number of objects connected with a field with a given name.

1.86.1 Detailed Description

The class representing the found physical page.

Definition at line 46 of file doc_physical_document.h.

1.86.2 Member Function Documentation

BasicObjectsBegin()

```
virtual DocBasicObjectsIterator se::doc::DocPhysicalPage::BasicObjectsBegin (
            const char * name ) const  [pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Returns a constant 'begin' iterator to the collection of basic objects connected with a field with a given name

1.87 se::doc::DocProcessingArguments Class Reference

The class representing the processing arguments for a custom document processor CLASS TO BE DEPRECATED.

```
#include <doc_external_processor.h>
```

Public Member Functions

- virtual ~**DocProcessingArguments** ()=default
  Default dtor.
- virtual int **GetTagArgumentsCount** () const =0
  Returns the number of arguments.
- virtual const char * **GetTagArgument** (int index) const =0
  Returns the argument by index.
- virtual void **SetTagArgument** (int index, const char *argument)=0
  Sets the argument by index.
- virtual void **Resize** (int size)=0
  Resizes the array of arguments.

1.87.1 Detailed Description

The class representing the processing arguments for a custom document processor CLASS TO BE DEPRECATED.

Definition at line 26 of file doc_external_processor.h.

1.88 se::doc::DocProcessingSettings Class Reference

The class representing the settings of a single processing iteration.

```
#include <doc_processing_settings.h>
```

Public Member Functions

- virtual ~**DocProcessingSettings** ()=default

  Default dtor.
- virtual int **GetOptionsCount** () const =0

  Returns the number of processing options.
- virtual bool **HasOption** (const char *option_name) const =0

  Returns true iff there exists a processing option with a given name.
- virtual const char * **GetOption** (const char *option_name) const =0

  Returns the processing option with a given name.
- virtual void **SetOption** (const char *option_name, const char *option_value)=0

  Sets the processing option as a key-value pair.
- virtual void **RemoveOption** (const char *option_name)=0

  Removes the processing option with a given name.
- virtual se::common::StringsMapIterator **OptionsBegin** () const =0

  Returns a 'begin' map-like iterator to the processing options.
- virtual se::common::StringsMapIterator **OptionsEnd** () const =0

  Returns an 'end' map-like iterator to the processing options.
- virtual int **GetSessionOptionsCount** () const =0

  Returns the number of session options.
- virtual bool **HasSessionOption** (const char *option_name) const =0

  Returns true iff there exists a session option with a given name.
- virtual const char * **GetSessionOption** (const char *option_name) const =0

  Returns the session option with a given name.
- virtual se::common::StringsMapIterator **SessionOptionsBegin** () const =0

  Returns a 'begin' map-like iterator to the session options.
- virtual se::common::StringsMapIterator **SessionOptionsEnd** () const =0

  Returns an 'end' map-like iterator to the session options.
- virtual int **GetEnabledDocumentTypesCount** () const =0

  Returns the number of enabled document types.
- virtual bool **HasEnabledDocumentType** (const char *doc_name) const =0

  Returns true iff there is an enabled document type with a given name.
- virtual const char * **GetEnabledDocumentType** (int doc_id) const =0

  Returns a name of enabled document type by index.
- virtual int GetCurrentSourceID () const =0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.
- virtual void **SetCurrentSourceID** (int source_id)=0

  Sets the ID of a view which is marked as a current source.
- virtual int **GetAvailableRoutinesCount** () const =0

  Returns the number of available routines.
- virtual bool **HasAvailableRoutine** (const char *routine_name) const =0

  Returns true iff there exists an available routine with a given name.
- virtual se::common::StringsMapIterator **AvailableRoutinesBegin** () const =0

  Returns a 'begin' map-like iterator to the list of routine names.
- virtual se::common::StringsMapIterator **AvailableRoutinesEnd** () const =0

  Returns an 'end' map-like iterator to the list of routine names.
- virtual int **RoutinesQueueSize** () const =0

  Returns the size of the current routines queue.
- virtual const char * **RoutinesQueueFront** () const =0

  Returns the routine name at the front of the queue.

- virtual void **RoutinesQueuePush** (const char *routine_name)=0

  Pushes a routine to the back of the current routines queue.
- virtual void **RoutinesQueuePop** ()=0

  Pops a routine from the front of the current routines queue.
- virtual void **RoutinesQueueClear** ()=0

  Cleas the routines queue.
- virtual void **BindFeedbackReporter** (DocFeedback *feedback_reporter)=0

  Binds feedback reporter to processing settings.
- virtual DocFeedback * **GetFeedbackReporter** () const =0

  Returns pointer to feedback reporter.

1.88.1 Detailed Description

The class representing the settings of a single processing iteration.

Definition at line 23 of file doc_processing_settings.h.

1.88.2 Member Function Documentation

GetCurrentSourceID()

```
virtual int se::doc::DocProcessingSettings::GetCurrentSourceID ( ) const  [pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Returns the ID of a view which is marked as a current source

1.89 se::doc::DocRawFieldFeedback Class Reference

The class representing a feedback for one raw field.

```
#include <doc_feedback.h>
```

Public Member Functions

- virtual ~**DocRawFieldFeedback** ()=default

  Default dtor.
- virtual const char * **GetName** () const =0

  Returns name of the raw field.
- virtual bool **HasQuadrangle** () const =0

  Returns true iff field has quadrangle.
- virtual const se::common::Quadrangle & **GetQuadrangle** () const =0

  Returns raw field quadrangle in the source page.
- virtual const char * **GetType** () const =0

  Returns type of the raw field.
- virtual const se::common::OcrString **GetOcrString** () const =0

  Returns recognized value of the raw field.

1.89.1 Detailed Description

The class representing a feedback for one raw field.

Definition at line 24 of file doc_feedback.h.

1.90 se::doc::DocRawFieldsFeedbackContainer Class Reference

The class representing a feedback container for raw fields.

`#include <doc_feedback.h>`

Public Member Functions

- virtual ~**DocRawFieldsFeedbackContainer** ()=default

  Default dtor.
- virtual int **GetRawFieldCount** () const =0

  Returns the number of raw fields.
- virtual int **GetSourcePageID** () const =0

  Returns ID of the source page.
- virtual const DocRawFieldFeedback & **GetRawFieldFeedback** (const int idx) const =0

  Return feedback for the raw field with given indice.

1.90.1 Detailed Description

The class representing a feedback container for raw fields.

Definition at line 49 of file doc_feedback.h.

1.91 se::doc::DocResult Class Reference

The class representing the document analysis and recognition result.

`#include <doc_result.h>`

Public Member Functions

- virtual ~**DocResult** ()=default

  Default dtor.
- virtual DocResult * **PartialClone** () const =0

  Returns DoctResult copy without graphical structure or physical document.
- virtual DocResult * **Clone** () const =0

  Returns DoctResult copy.
- virtual int **GetDocumentsCount** () const =0

  Returns the number of found documents.
- virtual bool **HasDocument** (int doc_id) const =0

  Returns true iff there is a document with a given ID.
- virtual const Document & **GetDocument** (int doc_id) const =0

  Returns a document with a given ID (const ref)
- virtual const Document * **GetDocumentPtr** (int doc_id) const =0

  Returns a document with a given ID (const ptr)
- virtual DocumentsIterator **DocumentsBegin** () const =0

  Returns a constant 'begin' iterator to the collection of documents.
- virtual DocumentsIterator **DocumentsEnd** () const =0

  Returns a constant 'end' iterator to the collection of documents.
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the result instance with a given serializer object.
- virtual const DocPhysicalDocument & **GetPhysicalDocument** (int idx) const =0

  Returns a physical document with a given indice (const ref)
- virtual const DocPhysicalDocument * **GetPhysicalDocumentPtr** (int idx) const =0

  Returns a physical document with a given indice (const ptr)
- virtual int **GetScenesCount** () const =0

  Returns a count of scenes.
- virtual const DocSceneInfo & **GetSceneInfo** (int idx) const =0

  Returns a scene info with a given indice (const ref)
- virtual const DocSceneInfo & **GetLastSceneInfo** () const =0

  Returns last scene info (const ref)
- virtual const DocSceneInfo * **GetSceneInfoPtr** (int idx) const =0

  Returns a scene info with a given indice (const ptr)
- virtual const DocSceneInfo * **GetLastSceneInfoPtr** () const =0

  Returns last scene info (const ptr)
- virtual const DocGraphicalStructure & GetGraphicalStructure () const =0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.
- virtual DocGraphicalStructure & **GetMutableGraphicalStructure** ()=0

  Returns the graphical structure of the analyzed images (mutable ref)
- virtual const DocGraphicalStructure * **GetGraphicalStructurePtr** () const =0

  Returns the graphical structure of the analyzed images (const ptr)
- virtual DocGraphicalStructure * **GetMutableGraphicalStructurePtr** ()=0

  Returns the graphical structure of the analyzed images (mutable ptr)
- virtual Document & **GetMutableDocument** (int doc_id)=0

  Returns a document with a given ID (mutable ref)
- virtual const DocTagsCollection & **GetDocumentTags** (int doc_id) const =0

  Returns the tags collection of a document with a given ID.
- virtual Document * **GetMutableDocumentPtr** (int doc_id)=0

  Returns a document with a given ID (mutable ref)

- virtual const DocTagsCollection * **GetDocumentTagsPtr** (int doc_id) const =0

  Returns the tags collection of a document with a given ID.
- virtual DocumentsMutableIterator **AddDocument** (const Document &doc)=0

  Adds a new document to the result.
- virtual void **SetDocument** (int doc_id, const Document &doc)=0

  Sets a document with a given ID.
- virtual void **RemoveDocument** (int doc_id)=0

  Removes a document with a given ID.
- virtual DocumentsMutableIterator **MutableDocumentsBegin** ()=0

  Returns a mutable 'begin' iterator to the collection of documents.
- virtual DocumentsMutableIterator **MutableDocumentsEnd** ()=0

  Returns a mutable 'end' iterator to the collection of documents.
- virtual DocumentsSliceIterator **DocumentsSlice** (const char *tag) const =0

  Returns a constant iterator to the subset of documents with a given tag.
- virtual DocumentsMutableSliceIterator **MutableDocumentsSlice** (const char *tag)=0

  Returns a mutable iterator to the subset of documents with a given tag.
- virtual bool **CanBuildPDFABuffer** () const =0

  Checks if pdf/a buffer can be created.
- virtual void **BuildPDFABuffer** ()=0

  Converts result to pdf/a buffer.
- virtual void **GetPDFABuffer** (unsigned char *output_buf, unsigned long long buf_size) const =0

  Returns the buffer with pdf/a result.
- virtual int **GetPDFABufferSize** () const =0

  Return the size of resulting buffer with pdf/a.
- virtual void **SetAddTextMode** (const char *mode_name)=0

  Sets the current mode of pdf/a serializing.
- virtual const char * **GetAddTextMode** () const =0

  Returns the current mode of pdf/a serializing.
- virtual bool **HasAddTextMode** (const char *mode_name) const =0

  Returns true if there is a supported mode of pdf/a serializing with a given name.
- virtual se::common::StringsVectorIterator **AddTextModesBegin** () const =0

  Returns a 'begin' vector-like iterator to the list of supported mode names.
- virtual se::common::StringsVectorIterator **AddTextModesEnd** () const =0

  Returns an 'end' vector-like iterator to the list of supported mode names.
- virtual void **SetTextTypeMode** (const char *mode_name)=0

  Sets the current mode of pdf/a serializing.
- virtual const char * **GetTextTypeMode** () const =0

  Returns the current mode of pdf/a serializing.
- virtual bool **HasTextTypeMode** (const char *mode_name) const =0

  Returns true if there is a supported mode of pdf/a serializing with a given name.
- virtual se::common::StringsVectorIterator **TextTypeModesBegin** () const =0

  Returns a 'begin' vector-like iterator to the list of supported mode names.
- virtual se::common::StringsVectorIterator **TextTypeModesEnd** () const =0

  Returns an 'end' vector-like iterator to the list of supported mode names.
- virtual void **SetColourMode** (const bool with_colour)=0

  Set to true if you want to save color elements of the images.
- virtual bool **GetColourMode** () const =0

  Return the current color saving mode.

1.91.1 Detailed Description

The class representing the document analysis and recognition result.

Definition at line 25 of file doc_result.h.

1.91.2 Member Function Documentation

GetGraphicalStructure()

```
virtual const DocGraphicalStructure & se::doc::DocResult::GetGraphicalStructure ( ) const
[pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Returns the graphical structure of the analyzed images (const ref)

1.92 se::doc::DocSceneInfo Class Reference

The class representing basic information about a scene.

`#include <doc_scene_info.h>`

Public Types

- enum class SceneOriginType

  Enumeration of possible image sources.

Public Member Functions

- virtual ~**DocSceneInfo** ()=default

  Default dtor.
- virtual bool **IsGarbage** () const =0

  Returns true iff the scene is invalid.
- virtual int **SceneID** () const =0

  Returns scene ID in the flow.
- virtual int **GarbageReasonsCount** () const =0

  Returns the number of garbage reasons for the scene.
- virtual const char * **GarbageReason** (int idx) const =0

  Returns the garbage reason with a given indice.
- virtual SceneOriginType **GetSceneOriginType** () const =0

  Returns the scene origin type.
- virtual int GetForensicCheckFieldsCount () const =0

  Results of scene-based forensic checks.
- virtual bool **HasForensicCheckField** (const char *name) const =0

  Checks if a forensic check field exists by name.
- virtual const DocForensicCheckField & **GetForensicCheckField** (const char *name) const =0

  Forensic check field getter by name.
- virtual const DocForensicCheckField * **GetForensicCheckFieldPtr** (const char *name) const =0

  Forensic check field getter by name.
- virtual DocForensicCheckFieldsIterator **ForensicCheckFieldsBegin** () const =0

  Returns a begin-iterator for an internal collection of forensic check fields.
- virtual DocForensicCheckFieldsIterator **ForensicCheckFieldsEnd** () const =0

  Returns an end-iterator for an internal collection of forensic check fields.

1.92.1 Detailed Description

The class representing basic information about a scene.

Definition at line 22 of file doc_scene_info.h.

1.92.2 Member Enumeration Documentation

SceneOriginType

```
enum class se::doc::DocSceneInfo::SceneOriginType  [strong]
```

Enumeration of possible image sources.

Definition at line 27 of file doc_scene_info.h.

1.92.3 Member Function Documentation

GetForensicCheckFieldsCount()

```
virtual int se::doc::DocSceneInfo::GetForensicCheckFieldsCount ( ) const  [pure virtual]
```

Results of scene-based forensic checks.

Returns the number of forensic check fields in a document

1.93 se::doc::DocSession Class Reference

The class representing image processing session - main processing class of Smart Document Engine.

```
#include <doc_session.h>
```

Public Member Functions

- virtual ~**DocSession** ()=default

  Default dtor.
- virtual DocProcessingSettings * CreateProcessingSettings () const =0

  Creates a processing settings instance.
- virtual const char * GetActivationRequest ()=0

  Get an activation request for this session (valid for SDK built with dynamic activation feature)
- virtual void Activate (const char *activation_response)=0

  Activate current session (valid for SDK built with dynamic activation feature)
- virtual bool IsActivated () const =0

  Check if current session was activated (valid for SDK built with dynamic activation feature)
- virtual void ProcessImage (const se::common::Image &in_image, const DocProcessingSettings *settings=nullptr)=0

  Processes an image.
- virtual void **Reset** ()=0

  Resets the internal state of the processing session.

- virtual const DocResult & **GetCurrentResult** () const =0

  Returns the current result (const ref)
- virtual const DocResult * **GetCurrentResultPtr** () const =0

  Returns the current result (const ptr)
- virtual const char * **GetType** () const =0

  Returns session type.
- virtual int RegisterImage (const se::common::Image &in_image)=0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.
- virtual void **Process** (DocProcessingSettings &settings)=0

  Launches the document processing iteration with given settings.
- virtual DocResult & **GetMutableCurrentResult** ()=0

  Returns the current result (mutable ref)
- virtual DocResult * **GetMutableCurrentResultPtr** ()=0

  Returns the current result (mutable ptr)

1.93.1 Detailed Description

The class representing image processing session - main processing class of Smart Document Engine.

Definition at line 24 of file doc_session.h.

1.93.2 Member Function Documentation

CreateProcessingSettings()

```
virtual DocProcessingSettings * se::doc::DocSession::CreateProcessingSettings ( ) const  [pure virtual]
```

Creates a processing settings instance.

Returns

A newly created processing settings instance. An object is allocated, the caller is responsible for deleting it.

GetActivationRequest()

```
virtual const char * se::doc::DocSession::GetActivationRequest ( )  [pure virtual]
```

Get an activation request for this session (valid for SDK built with dynamic activation feature)

Returns

A string with activation request

Activate()

```
virtual void se::doc::DocSession::Activate (
            const char * activation_response )  [pure virtual]
```

Activate current session (valid for SDK built with dynamic activation feature)

Parameters

activation_response	- the response from activation server

IsActivated()

```
virtual bool se::doc::DocSession::IsActivated ( ) const  [pure virtual]
```

Check if current session was activated (valid for SDK built with dynamic activation feature)

Returns

Boolean check (true/false)

ProcessImage()

```
virtual void se::doc::DocSession::ProcessImage (
            const se::common::Image & in_image,
            const DocProcessingSettings * settings = nullptr )  [pure virtual]
```

Processes an image.

Parameters

in_image	- the input image to process
settings	- DocProcessingSettings instance

RegisterImage()

```
virtual int se::doc::DocSession::RegisterImage (
            const se::common::Image & in_image )  [pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Registers a new image in the graphical structure

Parameters

in_image	- the input image to register

Returns

the ID of the view corresponding to the registered image

1.94 se::doc::DocSessionSettings Class Reference

The class representing the document processing session settings.

`#include <doc_session_settings.h>`

Public Member Functions

- virtual ~**DocSessionSettings** ()=default
  Default dtor.
- virtual DocSessionSettings * Clone () const =0
  Clones the session settings object.
- virtual int **GetOptionsCount** () const =0
  Returns the number of session options.
- virtual bool **HasOption** (const char *option_name) const =0
  Returns true iff there is a session option with a given name.
- virtual const char * **GetOption** (const char *option_name) const =0
  Returns the session option with a given name.
- virtual void **SetOption** (const char *option_name, const char *option_value)=0
  Sets a session option as a key-value pair.
- virtual void **RemoveOption** (const char *option_name)=0
  Removes the session option with a given name.
- virtual se::common::StringsMapIterator **OptionsBegin** () const =0
  Returns a 'begin' map-like iterator to the collection of session options.
- virtual se::common::StringsMapIterator **OptionsEnd** () const =0
  Returns an 'end' map-like iterator to the collection of session options.
- virtual int **GetSupportedModesCount** () const =0
  Returns the number of supported modes.
- virtual bool **HasSupportedMode** (const char *mode_name) const =0
  Returns true iff there is a supported mode with a given name.
- virtual const char * **GetSupportedMode** (int mode_id) const =0
  Returns the supported mode name with a given index.
- virtual se::common::StringsVectorIterator **SupportedModesBegin** () const =0
  Returns a 'begin' vector-like iterator to the list of supported mode names.
- virtual se::common::StringsVectorIterator **SupportedModesEnd** () const =0
  Returns an 'end' vector-like iterator to the list of supported mode names.
- virtual const char * **GetCurrentMode** () const =0
  Returns the current session mode.
- virtual void **SetCurrentMode** (const char *mode_name)=0
  Sets the current session mode.
- virtual int **GetInternalEnginesCount** () const =0
  Returns the number of available internal engines.
- virtual int **GetSupportedDocumentTypesCount** (int engine_id) const =0
  Returns the number of supported document types within an internal engin with a given index.
- virtual bool **HasSupportedDocumentType** (int engine_id, const char *doc_name) const =0
  Returns true iff there is a supported document type with a given name within an internal engine with a given index.

- virtual const char * **GetSupportedDocumentType** (int engine_id, int doc_id) const =0

  Returns the supported document type name with a given indices of the internal engine and document type.
- virtual int **GetEnabledDocumentTypesCount** () const =0

  Returns the number of enabled document types.
- virtual bool **HasEnabledDocumentType** (const char *doc_name) const =0

  Returns true iff there is an enabled document type with a given name.
- virtual const char * **GetEnabledDocumentType** (int doc_id) const =0

  Returns the enabled document type name with a given index.
- virtual const DocDocumentInfo & **GetDocumentInfo** (const char *doc_name) const =0

  Gets reference information about document type.
- virtual const DocDocumentInfo * **GetDocumentInfoPtr** (const char *doc_name) const =0

  Gets reference information about document type.
- virtual void AddEnabledDocumentTypes (const char *doc_type_mask)=0

  Adds enabled document types to the session settings, within the currently active mode.
- virtual void RemoveEnabledDocumentTypes (const char *doc_type_mask)=0

  Removes the document types from the set of enabled ones.
- virtual se::common::StringsSetIterator **PermissiblePrefixDocMasksBegin** ()=0

  Returns a 'begin' iterator to the set of permissible prefix document masks for the current mode.
- virtual se::common::StringsSetIterator **PermissiblePrefixDocMasksEnd** ()=0

  Returns an 'end' iterator to the set of permissible prefix document masks for the current mode.
- virtual bool **IsForensicsEnabled** () const =0

  Returns true iff the document forensics functionality is enabled.
- virtual void **EnableForensics** ()=0

  Enables the document forensics functionality.
- virtual void **DisableForensics** ()=0

  Disables the document forensics functionality.

1.94.1 Detailed Description

The class representing the document processing session settings.

Definition at line 24 of file doc_session_settings.h.

1.94.2 Member Function Documentation

Clone()

```
virtual DocSessionSettings * se::doc::DocSessionSettings::Clone ( ) const  [pure virtual]
```

Clones the session settings object.

Returns

A newly created object - deep copy of an instance. The object is allocated, the caller is responsible for deleting it.

AddEnabledDocumentTypes()

```
virtual void se::doc::DocSessionSettings::AddEnabledDocumentTypes (
            const char * doc_type_mask )  [pure virtual]
```

Adds enabled document types to the session settings, within the currently active mode.

Parameters

doc_type_mask	- a document type, or a mask with wildcards ('∗'). The wildcard symbol will match any sequence of characters, and the lookup dictionary for matched document types are taken from the set of supported document types within the currently active mode.

NB: the set of matched document types must belong to a single internal engine.

RemoveEnabledDocumentTypes()

```
virtual void se::doc::DocSessionSettings::RemoveEnabledDocumentTypes (
            const char * doc_type_mask )  [pure virtual]
```

Removes the document types from the set of enabled ones.

Parameters

doc_type_mask	- a document type, or a mask with wildcards ('∗'). The wildcard symbol will match any sequence of characters, and the lookup dictionary for matched document types are taken from the set of supported document types within the currently active mode.

1.95 se::doc::DocTableField Class Reference

The class representing a table field of a document.

```
#include <doc_fields.h>
```

Public Member Functions

- virtual ∼**DocTableField** ()=default

  Default dtor.
- virtual const DocBaseFieldInfo & **GetBaseFieldInfo** () const =0

  Returns the basic field information (const ref)
- virtual DocBaseFieldInfo & **GetMutableBaseFieldInfo** ()=0

  Returns the basic field information (mutable ref)
- virtual const DocBaseFieldInfo ∗ **GetBaseFieldInfoPtr** () const =0

  Returns the basic field information (const ptr)
- virtual DocBaseFieldInfo ∗ **GetMutableBaseFieldInfoPtr** ()=0

  Returns the basic field information (mutable ptr)
- virtual int **GetRowsCount** () const =0

  Returns the number of rows in the table.
- virtual int **GetColsCount** () const =0

  Returns the number of columns in the table.
- virtual const DocTextField & **GetCell** (int row, int col) const =0

  Returns the cell of a table as a DocTextField (const ref)
- virtual DocTextField & **GetMutableCell** (int row, int col)=0

  Returns the cell of a table as a DocTextField (mutable ref)
- virtual const DocTextField ∗ **GetCellPtr** (int row, int col) const =0

Returns the cell of a table as a DocTextField (const ptr)

- virtual DocTextField * **GetMutableCellPtr** (int row, int col)=0

  Returns the cell of a table as a DocTextField (mutable ptr)

- virtual void **SetCell** (int row, int col, const DocTextField &text_field)=0

  Sets the cell of a table as a DocTextField.

- virtual bool **HasColumnIndexByName** (const char *col_name) const =0

  Returns true if a column with given name exists, false, if not, or throws an exception if given column name is not in the documentation.

- virtual int **GetColumnIndexByName** (const char *col_name) const =0

  Returns index of the column with given name, or throws exception if given column is not in the table or in the documentation.

- virtual void **ResizeRows** (int rows)=0

  Resizes the set of rows of the table.

- virtual void **ResizeRows** (int rows, const DocTextField &filler)=0

  Resizes the set of rows of the table with a given filler cell value.

- virtual void **ResizeCols** (int cols)=0

  Resizes the set of columns of the table.

- virtual void **ResizeCols** (int cols, const DocTextField &filler)=0

  Resizes the set of columns of the table with a given filler cell value.

- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the field instance with a given serializer object.

- virtual int **GetHeaderRowsCount** () const =0

  Returns the number of rows in the table header.

- virtual int **GetHeaderColsCount** () const =0

  Returns the number of columns in the table header.

- virtual const DocTextField & **GetHeaderCell** (int row, int col) const =0

  Returns the cell of a table header as a DocTextField (const ref)

- virtual DocTextField & **GetHeaderMutableCell** (int row, int col)=0

  Returns the cell of a table header as a DocTextField (mutable ref)

- virtual const DocTextField * **GetHeaderCellPtr** (int row, int col) const =0

  Returns the cell of a table header as a DocTextField (const ptr)

- virtual DocTextField * **GetHeaderMutableCellPtr** (int row, int col)=0

  Returns the cell of a table header as a DocTextField (mutable ptr)

- virtual void **SetHeaderCell** (int row, int col, const DocTextField &text_field)=0

  Sets the cell of a table header as a DocTextField.

- virtual void **ResizeHeaderRows** (int rows)=0

  Resizes the set of rows of the table header.

- virtual void **ResizeHeaderRows** (int rows, const DocTextField &filler)=0

  Resizes the set of rows of the table header with a given filler cell value.

- virtual void **ResizeHeaderCols** (int cols)=0

  Resizes the set of columns of the table header.

- virtual void **ResizeHeaderCols** (int cols, const DocTextField &filler)=0

  Resizes the set of columns of the table header with a given filler cell value.

- virtual const char * GetColName (int col) const =0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

- virtual void **SetColName** (int col, const char *col_name)=0

  Sets the name of the column with a given index.

1.95.1 Detailed Description

The class representing a table field of a document.

Definition at line 296 of file doc_fields.h.

1.95.2 Member Function Documentation

GetColName()

```
virtual const char * se::doc::DocTableField::GetColName (
            int col ) const  [pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Returns the name of the column with a given index

1.96 se::doc::DocTableFieldsIterator Class Reference

Const-ref iterator for a collection of table fields.

```
#include <doc_fields_iterators.h>
```

Public Member Functions

- **DocTableFieldsIterator** (const DocTableFieldsIterator &other)
  Copy ctor.
- DocTableFieldsIterator & **operator=** (const DocTableFieldsIterator &other)
  Assignment operator.
- ~**DocTableFieldsIterator** ()
  Non-trivial dtor.
- const char * **GetKey** () const
  Returns the field name (the collection key)
- const DocTableField & **GetField** () const
  Returns the field value (const ref)
- const DocTableField * **GetFieldPtr** () const
  Returns the field value (const ptr)
- void **Advance** ()
  Switches the iterator to point on the next field in its collection.
- void **operator++** ()
  Switches the iterator to point on the next field in its collection.
- bool **Equals** (const DocTableFieldsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same field.
- bool **operator==** (const DocTableFieldsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same field.
- bool **operator!=** (const DocTableFieldsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the different fields.

Static Public Member Functions

- static DocTableFieldsIterator **ConstructFromImpl** (const DocTableFieldsIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocTableFieldsIterator** (const DocTableFieldsIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- class DocTableFieldsIteratorImpl * pimpl_

  Pointer to internal implementation.

1.96.1 Detailed Description

Const-ref iterator for a collection of table fields.

Definition at line 265 of file doc_fields_iterators.h.

1.96.2 Member Data Documentation

pimpl_

```
class DocTableFieldsIteratorImpl* se::doc::DocTableFieldsIterator::pimpl_  [private]
```

Pointer to internal implementation.

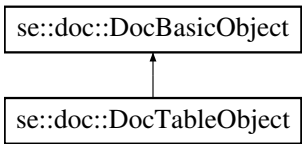
Definition at line 302 of file doc_fields_iterators.h.

1.97 se::doc::DocTableObject Class Reference

The graphical object representing a table.

```
#include <doc_objects.h>
```

Inheritance diagram for se::doc::DocTableObject:



Public Member Functions

- virtual ~**DocTableObject** () override=default

  Default dtor.
- virtual int **GetRowsCount** () const =0

  Returns the number of table rows.
- virtual int **GetColsCount** (int row) const =0

  Returns the number of table columns. Rows may contain different number of columns.
- virtual void **ResizeRows** (int rows)=0

  Resizes the set of rows.
- virtual void **ResizeCols** (int row, int cols)=0

  Resizes the set of columns.
- virtual const char * **GetColName** (int col, int row) const =0

  Returns the name of the column.
- virtual void **SetColName** (int col, int first_row, const char *col_name)=0

  Sets the name of the column. Number of columns in first row limits the number of column names.
- virtual const DocTextObject & **GetTextCell** (int row, int col) const =0

  Returns the text object of the given cell (const ref)
- virtual const DocTextObject * **GetTextCellPtr** (int row, int col) const =0

  Returns the text object of the given cell (const ptr)
- virtual const DocMultiStringTextObject & GetCell (int row, int col) const =0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.
- virtual DocMultiStringTextObject & **GetMutableCell** (int row, int col)=0

  Returns the cell by given row and column indices (mutable ref)
- virtual const DocMultiStringTextObject * **GetCellPtr** (int row, int col) const =0

  Returns the cell by given row and column indices (const ptr)
- virtual DocMultiStringTextObject * **GetMutableCellPtr** (int row, int col)=0

  Returns the cell by given row and column indices (mutable ptr)
- virtual void **SetCell** (int row, int col, const DocMultiStringTextObject &multi_string_text_object)=0

  Sets the cell by given row and column indices.

Public Member Functions inherited from se::doc::DocBasicObject

- virtual ~**DocBasicObject** ()=default

  Default dtor.
- virtual const char * **ObjectType** () const =0

  Returns the name of the concrete object type.
- virtual const DocBaseObjectInfo & **GetBaseObjectInfo** () const =0

  Returns the general basic object info (const ref)
- virtual DocBaseObjectInfo & **GetMutableBaseObjectInfo** ()=0

  Returns the general basic object info (mutable ref ref)
- virtual const DocBaseObjectInfo * **GetBaseObjectInfoPtr** () const =0

  Returns the general basic object info (const ptr)
- virtual DocBaseObjectInfo * **GetMutableBaseObjectInfoPtr** ()=0

  Returns the general basic object info (mutable ptr)
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the object instance with a given serializer object.

Static Public Member Functions

- static const char * **ObjectTypeStatic** ()

  Returns the object type name.

Static Public Member Functions inherited from se::doc::DocBasicObject

- static const char * **BaseClassNameStatic** ()

  Static class name method, returns 'DocBasicObject'.

1.97.1 Detailed Description

The graphical object representing a table.

Definition at line 258 of file doc_objects.h.

1.97.2 Member Function Documentation

GetCell()

```
virtual const DocMultiStringTextObject & se::doc::DocTableObject::GetCell (
            int row,
            int col ) const  [pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Returns the cell by given row and column indices (const ref)

1.98 se::doc::DocTableObjectsCrossPageIterator Class Reference

Basic const-ref iterator for a collection of table objects from several pages.

```
#include <doc_physical_document_iterators.h>
```

Public Member Functions

- **DocTableObjectsCrossPageIterator** (const DocTableObjectsCrossPageIterator &other)

  Copy ctor.
- DocTableObjectsCrossPageIterator & **operator=** (const DocTableObjectsCrossPageIterator &other)

  Assignment operator.
- ~**DocTableObjectsCrossPageIterator** ()

  Non-trivial dtor.
- int **GetPhysicalPageID** () const

  Return ID of a physical page contaning current object.
- int **GetObjectID** () const

  Return ID of an object.
- const DocTableObject & **GetTableObject** () const

  Returns the table object (const ref)
- const DocTableObject * **GetTableObjectPtr** () const

  Returns the table object (const ptr)
- void **Advance** ()

  Switches the iterator to point on the next object in its collection.
- bool **Equals** (const DocTableObjectsCrossPageIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocTableObjectsCrossPageIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocTableObjectsCrossPageIterator &rvalue) const

  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocTableObjectsCrossPageIterator **ConstructFromImpl** (const DocTableObjectsCrossPageIterator↩ Impl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocTableObjectsCrossPageIterator** (const DocTableObjectsCrossPageIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocTableObjectsCrossPageIteratorImpl * pimpl_

  Pointer to internal implementation.

1.98.1 Detailed Description

Basic const-ref iterator for a collection of table objects from several pages.

Definition at line 171 of file doc_physical_document_iterators.h.

1.98.2 Member Data Documentation

pimpl_

```
DocTableObjectsCrossPageIteratorImpl* se::doc::DocTableObjectsCrossPageIterator::pimpl_↩
[private]
```

Pointer to internal implementation.

Definition at line 210 of file doc_physical_document_iterators.h.

1.99 se::doc::DocTableObjectsIterator Class Reference

Public Member Functions

- **DocTableObjectsIterator** (const DocTableObjectsIterator &other)
  Copy ctor.
- DocTableObjectsIterator & **operator=** (const DocTableObjectsIterator &other)
  Assignment operator.
- ~**DocTableObjectsIterator** ()
  Non-trivial dtor.
- const DocTableObject & **GetTableObject** () const
  Returns the table object (const ref)
- const DocTableObject * **GetTableObjectPtr** () const
  Returns the table object (const ptr)
- void **Advance** ()
  Switches the iterator to point on the next object in its collection.
- bool **Equals** (const DocTableObjectsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocTableObjectsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocTableObjectsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocTableObjectsIterator **ConstructFromImpl** (const DocTableObjectsIteratorImpl &pimpl)
  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocTableObjectsIterator** (const DocTableObjectsIteratorImpl &pimpl)
  Private ctor from internal implementation.

Private Attributes

- DocTableObjectsIteratorImpl * pimpl_
  Pointer to internal implementation.

1.99.1 Detailed Description

Definition at line 194 of file doc_basic_objects_iterator.h.

1.99.2 Member Data Documentation

pimpl_

```
DocTableObjectsIteratorImpl* se::doc::DocTableObjectsIterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 227 of file doc_basic_objects_iterator.h.

1.100 se::doc::DocTagsCollection Class Reference

The class representing the collection of tags CLASS TO BE DEPRECATED.

```
#include <doc_tags_collection.h>
```

Public Member Functions

- virtual ~**DocTagsCollection** ()=default
  Default dtor.
- virtual int **GetTagsCount** () const =0
  Returns the number of tags.
- virtual bool **HasTag** (const char *tag) const =0
  Returns true iff there is a tag with a given name in the collection.
- virtual void **AddTag** (const char *tag)=0
  Adds a tag with a given name to the collection.
- virtual void **RemoveTag** (const char *tag)=0
  Removes a tag with a given name from the collection.
- virtual se::common::StringsSetIterator **TagsBegin** () const =0
  Returns a 'begin' set-like iterator to the collection.
- virtual se::common::StringsSetIterator **TagsEnd** () const =0
  Returns an 'end' set-like iterator to the collection.
- virtual void **Serialize** (se::common::Serializer &serializer) const =0
  Serializes the tags collection instance with a given serializer object.

Static Public Member Functions

- static DocTagsCollection * Create ()
  Creates a new DocTagsCollection object.

1.100.1 Detailed Description

The class representing the collection of tags CLASS TO BE DEPRECATED.

Definition at line 23 of file doc_tags_collection.h.

1.100.2 Member Function Documentation

Create()

```
static DocTagsCollection * se::doc::DocTagsCollection::Create ( )  [static]
```

Creates a new DocTagsCollection object.

Returns

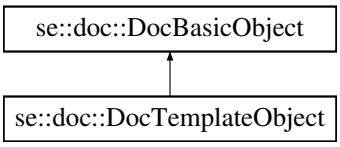
A newly created object with empty collection. The object is allocated, the caller is responsible for deleting it.

1.101 se::doc::DocTemplateObject Class Reference

The graphical object representing a fixed subform template.

```
#include <doc_objects.h>
```

Inheritance diagram for se::doc::DocTemplateObject:



Public Member Functions

- virtual ~**DocTemplateObject** () override=default
  Default dtor.

Public Member Functions inherited from se::doc::DocBasicObject

- virtual ~**DocBasicObject** ()=default
  Default dtor.
- virtual const char * **ObjectType** () const =0
  Returns the name of the concrete object type.
- virtual const DocBaseObjectInfo & **GetBaseObjectInfo** () const =0
  Returns the general basic object info (const ref)
- virtual DocBaseObjectInfo & **GetMutableBaseObjectInfo** ()=0
  Returns the general basic object info (mutable ref ref)
- virtual const DocBaseObjectInfo * **GetBaseObjectInfoPtr** () const =0
  Returns the general basic object info (const ptr)
- virtual DocBaseObjectInfo * **GetMutableBaseObjectInfoPtr** ()=0
  Returns the general basic object info (mutable ptr)
- virtual void **Serialize** (se::common::Serializer &serializer) const =0
  Serializes the object instance with a given serializer object.

Static Public Member Functions

- static const char * **ObjectTypeStatic** ()

  Returns the object type name.

Static Public Member Functions inherited from se::doc::DocBasicObject

- static const char * **BaseClassNameStatic** ()

  Static class name method, returns 'DocBasicObject'.

1.101.1 Detailed Description

The graphical object representing a fixed subform template.

Definition at line 146 of file doc_objects.h.

1.102 se::doc::DocTextField Class Reference

The class representing a text field of a document.

`#include <doc_fields.h>`

Public Member Functions

- virtual ~**DocTextField** ()=default

  Default dtor.
- virtual const DocBaseFieldInfo & **GetBaseFieldInfo** () const =0

  Returns the basic field information (const ref)
- virtual DocBaseFieldInfo & **GetMutableBaseFieldInfo** ()=0

  Returns the basic field information (mutable ref)
- virtual const DocBaseFieldInfo * **GetBaseFieldInfoPtr** () const =0

  Returns the basic field information (const ptr)
- virtual DocBaseFieldInfo * **GetMutableBaseFieldInfoPtr** ()=0

  Returns the basic field information (mutable ptr)
- virtual const se::common::OcrString & **GetOcrString** () const =0

  Returns the field recognition result (const ref)
- virtual se::common::OcrString & **GetMutableOcrString** ()=0

  Returns the field recognition result (mutable ref)
- virtual const se::common::OcrString * **GetOcrStringPtr** () const =0

  Returns the field recognition result (const ptr)
- virtual se::common::OcrString * **GetMutableOcrStringPtr** ()=0

  Returns the field recognition result (mutable ptr)
- virtual void **SetOcrString** (const se::common::OcrString &ocrstring)=0

  Sets the field recognition result.
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the field instance with a given serializer object.

1.102.1 Detailed Description

The class representing a text field of a document.

Definition at line 150 of file doc_fields.h.

1.103 se::doc::DocTextFieldsIterator Class Reference

Const-ref iterator for a collection of text fields.

`#include <doc_fields_iterators.h>`

Public Member Functions

- **DocTextFieldsIterator** (const DocTextFieldsIterator &other)
  Copy ctor.
- DocTextFieldsIterator & **operator=** (const DocTextFieldsIterator &other)
  Assignment operator.
- ~**DocTextFieldsIterator** ()
  Non-trivial dtor.
- const char * **GetKey** () const
  Returns the field name (the collection key)
- const DocTextField & **GetField** () const
  Returns the field value (const ref)
- const DocTextField * **GetFieldPtr** () const
  Returns the field value (const ptr)
- void **Advance** ()
  Switches the iterator to point on the next field in its collection.
- void **operator++** ()
  Switches the iterator to point on the next field in its collection.
- bool **Equals** (const DocTextFieldsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same field.
- bool **operator==** (const DocTextFieldsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same field.
- bool **operator!=** (const DocTextFieldsIterator &rvalue) const
  Returns true iff this instance and rvalue point to the different fields.

Static Public Member Functions

- static DocTextFieldsIterator **ConstructFromImpl** (const DocTextFieldsIteratorImpl &pimpl)
  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocTextFieldsIterator** (const DocTextFieldsIteratorImpl &pimpl)
  Private ctor from internal implementation.

Private Attributes

- class DocTextFieldsIteratorImpl * pimpl_
  Pointer to internal implementation.

1.103.1 Detailed Description

Const-ref iterator for a collection of text fields.

Definition at line 26 of file doc_fields_iterators.h.

1.103.2 Member Data Documentation

pimpl_

```
class DocTextFieldsIteratorImpl* se::doc::DocTextFieldsIterator::pimpl_  [private]
```

Pointer to internal implementation.

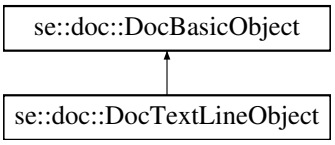
Definition at line 63 of file doc_fields_iterators.h.

1.104 se::doc::DocTextLineObject Class Reference

The graphical object representing a text line.

```
#include <doc_objects.h>
```

Inheritance diagram for se::doc::DocTextLineObject:



Public Member Functions

- virtual ~**DocTextLineObject** () override=default
  Default dtor.
- virtual const se::common::OcrString & **GetOcrString** () const =0
  Returns the text line recognition result (const ref)
- virtual const se::common::OcrString * **GetOcrStringPtr** () const =0
  Returns the text line recognition result (const ptr)
- virtual const se::common::QuadranglesVectorIterator **CellQuadranglesOnSceneBegin** () const =0
- virtual const se::common::QuadranglesVectorIterator **CellQuadranglesOnSceneEnd** () const =0
- virtual const se::common::QuadranglesVectorIterator **CellQuadranglesOnPageBegin** () const =0
- virtual const se::common::QuadranglesVectorIterator **CellQuadranglesOnPageEnd** () const =0
- virtual const se::common::Quadrangle & **GetCellQuadOnScene** (int idx) const =0
- virtual const se::common::Quadrangle & **GetCellQuadOnPage** (int idx) const =0

Public Member Functions inherited from se::doc::DocBasicObject

- virtual ~**DocBasicObject** ()=default

  Default dtor.
- virtual const char * **ObjectType** () const =0

  Returns the name of the concrete object type.
- virtual const DocBaseObjectInfo & **GetBaseObjectInfo** () const =0

  Returns the general basic object info (const ref)
- virtual DocBaseObjectInfo & **GetMutableBaseObjectInfo** ()=0

  Returns the general basic object info (mutable ref ref)
- virtual const DocBaseObjectInfo * **GetBaseObjectInfoPtr** () const =0

  Returns the general basic object info (const ptr)
- virtual DocBaseObjectInfo * **GetMutableBaseObjectInfoPtr** ()=0

  Returns the general basic object info (mutable ptr)
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the object instance with a given serializer object.

Static Public Member Functions

- static const char * **ObjectTypeStatic** ()

  Returns the object type name.

Static Public Member Functions inherited from se::doc::DocBasicObject

- static const char * **BaseClassNameStatic** ()

  Static class name method, returns 'DocBasicObject'.

1.104.1 Detailed Description

The graphical object representing a text line.

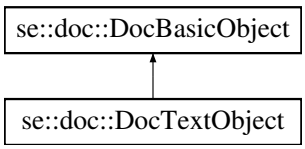
Definition at line 23 of file doc_objects.h.

1.105 se::doc::DocTextObject Class Reference

The graphical object representing a text.

`#include <doc_objects.h>`

Inheritance diagram for se::doc::DocTextObject:



Public Member Functions

- virtual ~**DocTextObject** () override=default

  Default dtor.
- virtual const se::common::OcrString & **GetOcrString** () const =0

  Returns the text line recognition result (const ref)
- virtual const se::common::OcrString * **GetOcrStringPtr** () const =0

  Returns the text line recognition result (const ptr)
- virtual int **GetTextLineObjectsCount** () const =0

  Return the number of text lines.
- virtual const DocTextLineObject & **GetTextLineObject** (int index) const =0

  Returns the text line object by line index (const ref)
- virtual const DocTextLineObject * **GetTextLineObjectPtr** (int index) const =0

  Returns the text line object by line index (const ptr)
- virtual se::common::OcrString & GetMutableOcrString ()=0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.
- virtual se::common::OcrString * **GetMutableOcrStringPtr** ()=0

  Returns the text line recognition result (mutable ptr)
- virtual void **SetOcrString** (const se::common::OcrString &ocrstring)=0

  Sets the text line recognition result.

Public Member Functions inherited from se::doc::DocBasicObject

- virtual ~**DocBasicObject** ()=default

  Default dtor.
- virtual const char * **ObjectType** () const =0

  Returns the name of the concrete object type.
- virtual const DocBaseObjectInfo & **GetBaseObjectInfo** () const =0

  Returns the general basic object info (const ref)
- virtual DocBaseObjectInfo & **GetMutableBaseObjectInfo** ()=0

  Returns the general basic object info (mutable ref ref)
- virtual const DocBaseObjectInfo * **GetBaseObjectInfoPtr** () const =0

  Returns the general basic object info (const ptr)
- virtual DocBaseObjectInfo * **GetMutableBaseObjectInfoPtr** ()=0

  Returns the general basic object info (mutable ptr)
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the object instance with a given serializer object.

Static Public Member Functions

- static const char * **ObjectTypeStatic** ()

  Returns the object type name.

Static Public Member Functions inherited from se::doc::DocBasicObject

- static const char * **BaseClassNameStatic** ()

  Static class name method, returns 'DocBasicObject'.

1.105.1 Detailed Description

The graphical object representing a text.

Definition at line 53 of file doc_objects.h.

1.105.2 Member Function Documentation

GetMutableOcrString()

```
virtual se::common::OcrString & se::doc::DocTextObject::GetMutableOcrString ( )  [pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Returns the text line recognition result (mutable ref)

1.106 se::doc::DocTextObjectsCrossPageIterator Class Reference

Basic const-ref iterator for a collection of text objects from several pages.

```
#include <doc_physical_document_iterators.h>
```

Public Member Functions

- **DocTextObjectsCrossPageIterator** (const DocTextObjectsCrossPageIterator &other)
  Copy ctor.
- DocTextObjectsCrossPageIterator & **operator=** (const DocTextObjectsCrossPageIterator &other)
  Assignment operator.
- ~**DocTextObjectsCrossPageIterator** ()
  Non-trivial dtor.
- int **GetPhysicalPageID** () const
  Return ID of a physical page contaning current object.
- int **GetObjectID** () const
  Return ID of an object.
- const DocTextObject & **GetTextObject** () const
  Returns the text object (const ref)
- const DocTextObject * **GetTextObjectPtr** () const
  Returns the text object (const ptr)
- void **Advance** ()
  Switches the iterator to point on the next object in its collection.
- bool **Equals** (const DocTextObjectsCrossPageIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocTextObjectsCrossPageIterator &rvalue) const
  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocTextObjectsCrossPageIterator &rvalue) const
  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocTextObjectsCrossPageIterator **ConstructFromImpl** (const DocTextObjectsCrossPageIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocTextObjectsCrossPageIterator** (const DocTextObjectsCrossPageIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocTextObjectsCrossPageIteratorImpl * pimpl_

  Pointer to internal implementation.

1.106.1 Detailed Description

Basic const-ref iterator for a collection of text objects from several pages.

Definition at line 36 of file doc_physical_document_iterators.h.

1.106.2 Member Data Documentation

pimpl_

```
DocTextObjectsCrossPageIteratorImpl* se::doc::DocTextObjectsCrossPageIterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 75 of file doc_physical_document_iterators.h.

1.107 se::doc::DocTextObjectsIterator Class Reference

Public Member Functions

- **DocTextObjectsIterator** (const DocTextObjectsIterator &other)

  Copy ctor.
- DocTextObjectsIterator & **operator=** (const DocTextObjectsIterator &other)

  Assignment operator.
- ~**DocTextObjectsIterator** ()

  Non-trivial dtor.
- const DocTextObject & **GetTextObject** () const

  Returns the text object (const ref)
- const DocTextObject * **GetTextObjectPtr** () const

  Returns the text object (const ptr)
- void **Advance** ()

  Switches the iterator to point on the next object in its collection.
- bool **Equals** (const DocTextObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator==** (const DocTextObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same object.
- bool **operator!=** (const DocTextObjectsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the different objects.

Static Public Member Functions

- static DocTextObjectsIterator **ConstructFromImpl** (const DocTextObjectsIteratorImpl &pimpl)
  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocTextObjectsIterator** (const DocTextObjectsIteratorImpl &pimpl)
  Private ctor from internal implementation.

Private Attributes

- DocTextObjectsIteratorImpl * pimpl_
  Pointer to internal implementation.

1.107.1 Detailed Description

Definition at line 85 of file doc_basic_objects_iterator.h.

1.107.2 Member Data Documentation

pimpl_

```
DocTextObjectsIteratorImpl* se::doc::DocTextObjectsIterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 118 of file doc_basic_objects_iterator.h.

1.108 se::doc::Document Class Reference

Class representing a recognized Document.

```
#include <doc_document.h>
```

Public Member Functions

- virtual ~**Document** ()=default

  Default dtor.
- virtual int **GetTextFieldsCount** () const =0

  Returns the number of text fields in a document.
- virtual bool **HasTextField** (const char *name) const =0

  Checks if a text field exists by name.
- virtual const DocTextField & **GetTextField** (const char *name) const =0

  Text field getter by name.
- virtual const DocTextField * **GetTextFieldPtr** (const char *name) const =0

  Text field getter by name.
- virtual DocTextFieldsIterator **TextFieldsBegin** () const =0

  Returns a begin-iterator for an internal collection of text fields.
- virtual DocTextFieldsIterator **TextFieldsEnd** () const =0

  Returns an end-iterator for an internal collection of text fields.
- virtual int **GetImageFieldsCount** () const =0

  Returns the number of image fields in a document.
- virtual bool **HasImageField** (const char *name) const =0

  Checks if an image field exists by name.
- virtual const DocImageField & **GetImageField** (const char *name) const =0

  Image field getter by name.
- virtual const DocImageField * **GetImageFieldPtr** (const char *name) const =0

  Image field getter by name.
- virtual DocImageFieldsIterator **ImageFieldsBegin** () const =0

  Returns a begin-iterator for an internal collection of image fields.
- virtual DocImageFieldsIterator **ImageFieldsEnd** () const =0

  Returns an end-iterator for an internal collection of image fields.
- virtual int **GetCheckboxFieldsCount** () const =0

  Returns the number of checkbox fields in a document.
- virtual bool **HasCheckboxField** (const char *name) const =0

  Checks if a checkbox field exists by name.
- virtual const DocCheckboxField & **GetCheckboxField** (const char *name) const =0

  Checkbox field getter by name.
- virtual const DocCheckboxField * **GetCheckboxFieldPtr** (const char *name) const =0

  Checkbox field getter by name.
- virtual DocCheckboxFieldsIterator **CheckboxFieldsBegin** () const =0

  Returns a begin-iterator for an internal collection of checkbox fields.
- virtual DocCheckboxFieldsIterator **CheckboxFieldsEnd** () const =0

  Returns an end-iterator for an internal collection of checkbox fields.
- virtual int **GetForensicFieldsCount** () const =0

  Returns the number of forensic fields in a document.
- virtual bool **HasForensicField** (const char *name) const =0

  Checks if a forensic field exists by name.
- virtual const DocForensicField & **GetForensicField** (const char *name) const =0

  Forensic field getter by name.
- virtual const DocForensicField * **GetForensicFieldPtr** (const char *name) const =0

  Forensic field getter by name.
- virtual DocForensicFieldsIterator **ForensicFieldsBegin** () const =0

  Returns a begin-iterator for an internal collection of forensic fields.
- virtual DocForensicFieldsIterator **ForensicFieldsEnd** () const =0

Returns an end-iterator for an internal collection of forensic fields.

- virtual int **GetForensicCheckFieldsCount** () const =0

  Returns the number of forensic check fields in a document.
- virtual bool **HasForensicCheckField** (const char ∗name) const =0

  Checks if a forensic check field exists by name.
- virtual const DocForensicCheckField & **GetForensicCheckField** (const char ∗name) const =0

  Forensic check field getter by name.
- virtual const DocForensicCheckField ∗ **GetForensicCheckFieldPtr** (const char ∗name) const =0

  Forensic check field getter by name.
- virtual DocForensicCheckFieldsIterator **ForensicCheckFieldsBegin** () const =0

  Returns a begin-iterator for an internal collection of forensic check fields.
- virtual DocForensicCheckFieldsIterator **ForensicCheckFieldsEnd** () const =0

  Returns an end-iterator for an internal collection of forensic check fields.
- virtual int **GetTableFieldsCount** () const =0

  Returns the number of table fields in a document.
- virtual bool **HasTableField** (const char ∗name) const =0

  Checks if a table field exists by name.
- virtual const DocTableField & **GetTableField** (const char ∗name) const =0

  Table field getter by name.
- virtual const DocTableField ∗ **GetTableFieldPtr** (const char ∗name) const =0

  Table field getter by name.
- virtual DocTableFieldsIterator **TableFieldsBegin** () const =0

  Returns a begin-iterator for an internal collection of table fields.
- virtual DocTableFieldsIterator **TableFieldsEnd** () const =0

  Returns an end-iterator for an internal collection of table fields.
- virtual int **GetBarcodeFieldsCount** () const =0

  Returns the number of barcode fields in a document.
- virtual bool **HasBarcodeField** (const char ∗name) const =0

  Checks if a barcode field exists by name.
- virtual const DocBarcodeField & **GetBarcodeField** (const char ∗name) const =0

  Barcode field getter by name.
- virtual const DocBarcodeField ∗ **GetBarcodeFieldPtr** (const char ∗name) const =0

  Barcode field getter by name.
- virtual DocBarcodeFieldsIterator **BarcodeFieldsBegin** () const =0

  Returns a begin-iterator for an internal collection of barcode fields.
- virtual DocBarcodeFieldsIterator **BarcodeFieldsEnd** () const =0

  Returns an end-iterator for an internal collection of barcode fields.
- virtual int **GetAttributesCount** () const =0

  Returns the number of document attributes.
- virtual bool **HasAttribute** (const char ∗attr_name) const =0

  Checks if an attributes exists with a given name.
- virtual const char ∗ **GetAttribute** (const char ∗attr_name) const =0

  Returns an attribute value for a given name.
- virtual void **SetAttribute** (const char ∗attr_name, const char ∗attr_value)=0

  Sets an attribute key-value pair.
- virtual void **RemoveAttribute** (const char ∗attr_name)=0

  Removes an attribute with a given name.
- virtual se::common::StringsMapIterator **AttributesBegin** () const =0

  Returns a begin-iterator for an internal collection of attributes.
- virtual se::common::StringsMapIterator **AttributesEnd** () const =0

  Returns an end-iterator for an internal collection of attributes.

- virtual const char * **GetType** () const =0

  Returns document's type.
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the document instance with a given serializer object.
- virtual int **GetPhysicalDocIdx** () const =0

  Return indice of the connected physical document.
- virtual DocTextField & GetMutableTextField (const char *name)=0

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.
- virtual DocTextField * **GetMutableTextFieldPtr** (const char *name)=0

  Mutable text field getter by name.
- virtual void **SetTextField** (const char *name, const DocTextField &field)=0

  Text field setter.
- virtual void **RemoveTextField** (const char *name)=0

  Removes a text field with a given name.
- virtual DocImageField & **GetMutableImageField** (const char *name)=0

  Mutable image field getter by name.
- virtual DocImageField * **GetMutableImageFieldPtr** (const char *name)=0

  Mutable image field getter by name.
- virtual void **SetImageField** (const char *name, const DocImageField &field)=0

  Image field setter.
- virtual void **RemoveImageField** (const char *name)=0

  Removes an image field with a given name.
- virtual DocCheckboxField & **GetMutableCheckboxField** (const char *name)=0

  Mutable checkbox field getter by name.
- virtual DocCheckboxField * **GetMutableCheckboxFieldPtr** (const char *name)=0

  Mutable checkbox field getter by name.
- virtual void **SetCheckboxField** (const char *name, const DocCheckboxField &field)=0

  Checkbox field setter.
- virtual void **RemoveCheckboxField** (const char *name)=0

  Removes a checkbox field with a given name.
- virtual DocForensicField & **GetMutableForensicField** (const char *name)=0

  Mutable forensic field getter by name.
- virtual DocForensicField * **GetMutableForensicFieldPtr** (const char *name)=0

  Mutable forensic field getter by name.
- virtual void **SetForensicField** (const char *name, const DocForensicField &field)=0

  Forensic field setter.
- virtual void **RemoveForensicField** (const char *name)=0

  Removes a forensic field with a given name.
- virtual DocForensicCheckField & **GetMutableForensicCheckField** (const char *name)=0

  Mutable forensic check field getter by name.
- virtual DocForensicCheckField * **GetMutableForensicCheckFieldPtr** (const char *name)=0

  Mutable forensic check field getter by name.
- virtual void **SetForensicCheckField** (const char *name, const DocForensicCheckField &field)=0

  Forensic check field setter.
- virtual void **RemoveForensicCheckField** (const char *name)=0

  Removes a forensic check field with a given name.
- virtual DocTableField & **GetMutableTableField** (const char *name)=0

  Mutable table field getter by name.
- virtual DocTableField * **GetMutableTableFieldPtr** (const char *name)=0

  Mutable table field getter by name.

- virtual void **SetTableField** (const char ∗name, const DocTableField &field)=0

  Table field setter.
- virtual void **RemoveTableField** (const char ∗name)=0

  Removes a table field with a given name.
- virtual DocBarcodeField & **GetMutableBarcodeField** (const char ∗name)=0

  Mutable barcode field getter by name.
- virtual DocBarcodeField ∗ **GetMutableBarcodeFieldPtr** (const char ∗name)=0

  Mutable barcode field getter by name.
- virtual void **SetBarcodeField** (const char ∗name, const DocBarcodeField &field)=0

  Barcode field setter.
- virtual void **RemoveBarcodeField** (const char ∗name)=0

  Removes a barcode field with a given name.

Static Public Member Functions

- static const char ∗ **BaseClassNameStatic** ()

  Service method, returns "Document".

1.108.1 Detailed Description

Class representing a recognized Document.

Definition at line 22 of file doc_document.h.

1.108.2 Member Function Documentation

GetMutableTextField()

```
virtual DocTextField & se::doc::Document::GetMutableTextField (
            const char * name )  [pure virtual]
```

METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS.

Mutable text field getter by name

1.109 se::doc::DocumentsIterator Class Reference

A constant iterator for a collection of Document instances.

```
#include <doc_documents_iterator.h>
```

Public Member Functions

- **DocumentsIterator** (const DocumentsIterator &other)

  Copy ctor.
- DocumentsIterator & **operator=** (const DocumentsIterator &other)

  Assignment operator.
- ∼**DocumentsIterator** ()

  Dtor (non-trivial)
- int **GetID** () const

  Returns a document ID.
- const Document & **GetDocument** () const

  Constant getter for the document instance.
- const Document ∗ **GetDocumentPtr** () const

  Constant getter for the document instance.
- void **Advance** ()

  Moves the iterator to the next object in a collection.
- void **operator++** ()

  Operator which moves the iterator to the next object in a collection.
- bool **Equals** (const DocumentsIterator &rvalue) const

  Checks whether the iterator points to the same object as rvalue.
- bool **operator==** (const DocumentsIterator &rvalue) const

  Operator which checks whether the iterator points to the same object.
- bool **operator!=** (const DocumentsIterator &rvalue) const

  Operator which checks whether the iterator points to a different object.
- const DocTagsCollection & **GetTags** () const

  METHODS TO BE DEPRECATED THESE METHODS ARE A PART OF THE OLD INTERFACE THEY ARE TO BE DELETED IN FUTURE VERSIONS Returns a collection of tags associated with a document in the collection.
- const DocTagsCollection ∗ **GetTagsPtr** () const

  Returns a collection of tags associated with a document in the collection.

Static Public Member Functions

- static DocumentsIterator **ConstructFromImpl** (const DocumentsIteratorImpl &pimpl)

  A public handle for the internal ctor.

Private Member Functions

- **DocumentsIterator** (const DocumentsIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocumentsIteratorImpl ∗ pimpl_

  Internal representation of the iterator.

1.109.1 Detailed Description

A constant iterator for a collection of Document instances.

Definition at line 26 of file doc_documents_iterator.h.

1.109.2 Member Data Documentation

pimpl_

```
DocumentsIteratorImpl* se::doc::DocumentsIterator::pimpl_  [private]
```

Internal representation of the iterator.

Definition at line 71 of file doc_documents_iterator.h.

1.110 se::doc::DocumentsMutableIterator Class Reference

A mutable iterator for a collection of Document instances CLASS TO BE DEPRECATED.

```
#include <doc_documents_iterator.h>
```

Public Member Functions

- **DocumentsMutableIterator** (const DocumentsMutableIterator &other)
  Copy ctor.
- DocumentsMutableIterator & **operator=** (const DocumentsMutableIterator &other)
  Assignment operator.
- ~**DocumentsMutableIterator** ()
  Dtor (non-trivial)
- int **GetID** () const
  Returns a document ID.
- const Document & **GetDocument** () const
  Constant getter for the document instance.
- Document & **GetMutableDocument** () const
  Mutable getter for the document instance.
- const Document * **GetDocumentPtr** () const
  Constant getter for the document instance.
- Document * **GetMutableDocumentPtr** () const
  Mutable getter for the document instance.
- void **Advance** ()
  Moves the iterator to the next object in a collection.
- void **operator++** ()
  Operator which moves the iterator to the next object in a collection.
- bool **Equals** (const DocumentsMutableIterator &rvalue) const
  Checks whether the iterator points to the same object as rvalue.
- bool **operator==** (const DocumentsMutableIterator &rvalue) const
  Operator which checks whether the iterator points to the same object.
- bool **operator!=** (const DocumentsMutableIterator &rvalue) const
  Operator which checks whether the iterator points to a different object.
- const DocTagsCollection & **GetTags** () const
  Returns a collection of tags associated with a document in the collection.
- const DocTagsCollection * **GetTagsPtr** () const
  Returns a collection of tags associated with a document in the collection.

Static Public Member Functions

- static DocumentsMutableIterator **ConstructFromImpl** (const DocumentsMutableIteratorImpl &pimpl)
  A public handle for the main ctor.

Private Member Functions

- **DocumentsMutableIterator** (const DocumentsMutableIteratorImpl &pimpl)
  Private ctor from internal implementation.

Private Attributes

- DocumentsMutableIteratorImpl * pimpl_
  Internal representation of the iterator.

1.110.1 Detailed Description

A mutable iterator for a collection of Document instances CLASS TO BE DEPRECATED.

Definition at line 82 of file doc_documents_iterator.h.

1.110.2 Member Data Documentation

pimpl_

```
DocumentsMutableIteratorImpl* se::doc::DocumentsMutableIterator::pimpl_  [private]
```

Internal representation of the iterator.

Definition at line 128 of file doc_documents_iterator.h.

1.111 se::doc::DocumentsMutableSliceIterator Class Reference

A mutable iterator for a subset of the collection of Document instances TO BE DEPRECATED.

```
#include <doc_documents_iterator.h>
```

Public Member Functions

- **DocumentsMutableSliceIterator** (const DocumentsMutableSliceIterator &other)

  Copy ctor.
- DocumentsMutableSliceIterator & **operator=** (const DocumentsMutableSliceIterator &other)

  Assignment operator.
- ~**DocumentsMutableSliceIterator** ()

  Dtor (non-trivial)
- int **GetID** () const

  Returns a document ID.
- const Document & **GetDocument** () const

  Constant getter for the document instance.
- Document & **GetMutableDocument** () const

  Mutable getter for the document instance.
- const DocTagsCollection & **GetTags** () const

  Returns a collection of tags associated with a document in the collection.
- const Document * **GetDocumentPtr** () const

  Constant getter for the document instance.
- Document * **GetMutableDocumentPtr** () const

  Mutable getter for the document instance.
- const DocTagsCollection * **GetTagsPtr** () const

  Returns a collection of tags associated with a document in the collection.
- void **Advance** ()

  Moves the iterator to the next object in a collection.
- void **operator++** ()

  Operator which moves the iterator to the next object in a collection.
- bool **Finished** () const

  Returns true when the iterator reached the end of the subset.

Static Public Member Functions

- static DocumentsMutableSliceIterator **ConstructFromImpl** (const DocumentsMutableSliceIteratorImpl &pimpl)

  A public handle for the main ctor.

Private Member Functions

- **DocumentsMutableSliceIterator** (const DocumentsMutableSliceIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocumentsMutableSliceIteratorImpl * pimpl_

  Internal representation of the iterator.

1.111.1 Detailed Description

A mutable iterator for a subset of the collection of Document instances TO BE DEPRECATED.

Definition at line 194 of file doc_documents_iterator.h.

1.111.2 Member Data Documentation

pimpl_

```
DocumentsMutableSliceIteratorImpl* se::doc::DocumentsMutableSliceIterator::pimpl_  [private]
```

Internal representation of the iterator.

Definition at line 236 of file doc_documents_iterator.h.

1.112 se::doc::DocumentsSliceIterator Class Reference

A const iterator for a subset of the collection of Document instances TO BE DEPRECATED.

```
#include <doc_documents_iterator.h>
```

Public Member Functions

- **DocumentsSliceIterator** (const DocumentsSliceIterator &other)
  Copy ctor.
- DocumentsSliceIterator & **operator=** (const DocumentsSliceIterator &other)
  Assignment operator.
- ~**DocumentsSliceIterator** ()
  Dtor (non-trivial)
- int **GetID** () const
  Returns a document ID.
- const Document & **GetDocument** () const
  Constant getter for the document instance.
- const DocTagsCollection & **GetTags** () const
  Returns a collection of tags associated with a document in the collection.
- const Document * **GetDocumentPtr** () const
  Constant getter for the document instance.
- const DocTagsCollection * **GetTagsPtr** () const
  Returns a collection of tags associated with a document in the collection.
- void **Advance** ()
  Moves the iterator to the next object in a collection.
- void **operator++** ()
  Operator which moves the iterator to the next object in a collection.
- bool **Finished** () const
  Returns true when the iterator reached the end of the subset.

Static Public Member Functions

- static DocumentsSliceIterator **ConstructFromImpl** (const DocumentsSliceIteratorImpl &pimpl)
  A public handle for the main ctor.

Private Member Functions

- **DocumentsSliceIterator** (const DocumentsSliceIteratorImpl &pimpl)
  Private ctor from internal implementation.

Private Attributes

- DocumentsSliceIteratorImpl * pimpl_
  Internal representation of the iterator.

1.112.1 Detailed Description

A const iterator for a subset of the collection of Document instances TO BE DEPRECATED.

Definition at line 144 of file doc_documents_iterator.h.

1.112.2 Member Data Documentation

pimpl_

```
DocumentsSliceIteratorImpl* se::doc::DocumentsSliceIterator::pimpl_  [private]
```

Internal representation of the iterator.

Definition at line 181 of file doc_documents_iterator.h.

1.113 se::doc::DocVideoSession Class Reference

The class representing video processing session CLASS TO BE DEPRECATED.

```
#include <doc_video_session.h>
```

Public Member Functions

- virtual ~**DocVideoSession** ()=default
  Default dtor.
- virtual DocProcessingSettings * CreateProcessingSettings () const =0
  Creates a processing settings instance.
- virtual const char * GetActivationRequest ()=0
  Get an activation request for this session (valid for SDK built with dynamic activation feature)
- virtual void Activate (const char *activation_response)=0
  Activate current session (valid for SDK built with dynamic activation feature)
- virtual bool IsActivated () const =0
  Check if current session was activated (valid for SDK built with dynamic activation feature)
- virtual void ProcessImage (const se::common::Image &in_image, const DocProcessingSettings &settings)=0
  Launches processing of a video frame with given processing settings.
- virtual void **Reset** ()=0
  Resets the internal state of the session.
- virtual const DocResult & **GetCurrentResult** () const =0
  Returns the current result (const ref)
- virtual DocResult & **GetMutableCurrentResult** ()=0
  Returns the current result (mutable ref)
- virtual const DocResult * **GetCurrentResultPtr** () const =0
  Returns the current result (const ptr)
- virtual DocResult * **GetMutableCurrentResultPtr** ()=0
  Returns the current result (mutable ptr)

1.113.1 Detailed Description

The class representing video processing session CLASS TO BE DEPRECATED.

Definition at line 24 of file doc_video_session.h.

1.113.2 Member Function Documentation

CreateProcessingSettings()

```
virtual DocProcessingSettings * se::doc::DocVideoSession::CreateProcessingSettings ( ) const
[pure virtual]
```

Creates a processing settings instance.

Returns

A newly created processing settings instance. An object is allocated, the caller is responsible for deleting it.

GetActivationRequest()

```
virtual const char * se::doc::DocVideoSession::GetActivationRequest ( )  [pure virtual]
```

Get an activation request for this session (valid for SDK built with dynamic activation feature)

Returns

A string with activation request

Activate()

```
virtual void se::doc::DocVideoSession::Activate (
            const char * activation_response )  [pure virtual]
```

Activate current session (valid for SDK built with dynamic activation feature)

Parameters

activation_response	- the response from activation server

IsActivated()

```
virtual bool se::doc::DocVideoSession::IsActivated ( ) const  [pure virtual]
```

Check if current session was activated (valid for SDK built with dynamic activation feature)

Returns

Boolen check (true/false)

ProcessImage()

```
virtual void se::doc::DocVideoSession::ProcessImage (
            const se::common::Image & in_image,
            const DocProcessingSettings & settings )  [pure virtual]
```

Launches processing of a video frame with given processing settings.

Parameters

in_image	- input image for processing
settings	- processing settings instance

1.114 se::doc::DocView Class Reference

The class representing an image view stored in the graphical structure CLASS TO BE DEPRECATED.

```
#include <doc_view.h>
```

Public Member Functions

- virtual ~**DocView** ()=default

  Default dtor.
- virtual const se::common::Image & **GetImage** () const =0

  Returns the associated image (const ref)
- virtual se::common::Image & **GetMutableImage** ()=0

  Returns the associated image (mutable ref)
- virtual const se::common::Image * **GetImagePtr** () const =0

  Returns the associated image (const ptr)
- virtual se::common::Image * **GetMutableImagePtr** ()=0

  Returns the associated image (mutable ptr)
- virtual void **SetImage** (const se::common::Image &image)=0

  Sets the associated image.
- virtual int **GetAncestorID** () const =0

  Returns the immediate ancestor view ID in the views tree.
- virtual void **SetAncestorID** (int anc_id)=0

  Sets the immediate ancestor view ID in the views tree.
- virtual int **GetRootAncestorID** () const =0

  Returns the highest ancestor view ID in the views tree.
- virtual void **SetRootAncestorID** (int root_anc_id)=0

  Sets the highest ancestor view ID in the views tree.
- virtual const se::common::ProjectiveTransform & **GetTransform** () const =0

  Returns the projective transform from immediate ancestor to the current view (const ref)
- virtual se::common::ProjectiveTransform & **GetMutableTransform** ()=0

Returns the projective transform from immediate ancestor to the current view (mutable ref)

- virtual void **SetTransform** (const se::common::ProjectiveTransform &transform)=0

  Sets the projective transform from immediate ancestor to the current view.
- virtual const se::common::ProjectiveTransform * **GetTransformPtr** () const =0

  Returns the projective transform from immediate ancestor to the current view (const ptr)
- virtual se::common::ProjectiveTransform * **GetMutableTransformPtr** ()=0

  Returns the projective transform from immediate ancestor to the current view (mutable ptr)
- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the view instance with a given serializer object.

Static Public Member Functions

- static const char * **BaseClassNameStatic** ()

  Service method, returns object class name.

1.114.1 Detailed Description

The class representing an image view stored in the graphical structure CLASS TO BE DEPRECATED.

Definition at line 24 of file doc_view.h.

1.115 se::doc::DocViewsCollection Class Reference

The class representing the collection of views CLASS TO BE DEPRECATED.

`#include <doc_views_collection.h>`

Public Member Functions

- virtual ~**DocViewsCollection** ()=default

  Default dtor.
- virtual int **GetViewsCount** () const =0

  Returns the number of views.
- virtual bool **HasView** (int view_id) const =0

  Returns true iff there is a view with a given ID.
- virtual const DocView & **GetView** (int view_id) const =0

  Returns the view with a given ID (const ref)
- virtual DocView & **GetMutableView** (int view_id)=0

  Returns the view with a given ID (mutable ref)
- virtual const DocTagsCollection & **GetViewTags** (int view_id) const =0

  Returns the tags collection of the view with a given ID.
- virtual const DocView * **GetViewPtr** (int view_id) const =0

  Returns the view with a given ID (const ptr)
- virtual DocView * **GetMutableViewPtr** (int view_id)=0

  Returns the view with a given ID (mutable ptr)
- virtual const DocTagsCollection * **GetViewTagsPtr** (int view_id) const =0

  Returns the tags collection of the view with a given ID.
- virtual DocViewsMutableIterator RegisterView (const se::common::Image &image)=0

Registers a new top-level view.

- virtual DocViewsMutableIterator RegisterDerivedView (const se::common::Image &image, int ancestor_id, const se::common::ProjectiveTransform &transform)=0

  Registers a new derived view.

- virtual void **DeleteOrphans** ()=0

  Removes all views whose immediate ancestors do not exist.

- virtual void **DeleteView** (int view_id)=0

  Removes the view with a given ID.

- virtual DocViewsIterator **ViewsBegin** () const =0

  Returns a constant 'begin' iterator to the views collection.

- virtual DocViewsIterator **ViewsEnd** () const =0

  Returns a constant 'end' iterator to the views collection.

- virtual DocViewsMutableIterator **MutableViewsBegin** ()=0

  Returns a mutable 'begin' iterator to the views collection.

- virtual DocViewsMutableIterator **MutableViewsEnd** ()=0

  Returns a mutable 'end' iterator to the views collection.

- virtual DocViewsSliceIterator **ViewsSlice** (const char *tag) const =0

  Returns a constant iterator to the subset of views with a given tag.

- virtual DocViewsMutableSliceIterator **MutableViewsSlice** (const char *tag)=0

  Returns a mutable iterator to the subset of views with a given tag.

- virtual void **Serialize** (se::common::Serializer &serializer) const =0

  Serializes the instance with a given serializer object.

Static Public Member Functions

- static const char * **BaseClassNameStatic** ()

  Service method, returns object class name.

1.115.1 Detailed Description

The class representing the collection of views CLASS TO BE DEPRECATED.

Definition at line 25 of file doc_views_collection.h.

1.115.2 Member Function Documentation

RegisterView()

```
virtual DocViewsMutableIterator se::doc::DocViewsCollection::RegisterView (
            const se::common::Image & image )  [pure virtual]
```

Registers a new top-level view.

Parameters

image	- the image to associate with a new view

Returns

A mutable views iterator pointing to the newly created view

RegisterDerivedView()

```
virtual DocViewsMutableIterator se::doc::DocViewsCollection::RegisterDerivedView (
            const se::common::Image & image,
            int ancestor_id,
            const se::common::ProjectiveTransform & transform )  [pure virtual]
```

Registers a new derived view.

Parameters

image	- the image to associate with a new view
ancestor↩ _id	- the ID of an immediate ancestor
transform	- the projective tranform from immediate ancestor to the new view

Returns

A mutable views iterator pointing to the newly created view

1.116 se::doc::DocViewsIterator Class Reference

Basic const-ref iterator for a collection of views CLASS TO BE DEPRECATED.

`#include <doc_views_iterator.h>`

Public Member Functions

- **DocViewsIterator** (const DocViewsIterator &other)

  Copy ctor.
- DocViewsIterator & **operator=** (const DocViewsIterator &other)

  Assignment operator.
- ∼**DocViewsIterator** ()

  Non-trivial dtor.
- int **GetID** () const

  Returns the view ID.
- const DocView & **GetView** () const

  Returns the view (const ref)
- const DocTagsCollection & **GetTags** () const

  Returns the collection of tags associated with this view.
- const DocView ∗ **GetViewPtr** () const

  Returns the view (const ptr)
- const DocTagsCollection ∗ **GetTagsPtr** () const

  Returns the collection of tags associated with this view.
- void **Advance** ()

Switches the iterator to point on the next view.

- bool **Equals** (const DocViewsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same view.
- bool **operator==** (const DocViewsIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same view.
- bool **operator!=** (const DocViewsIterator &rvalue) const

  Returns true iff this instance and rvalue point to different views.

Static Public Member Functions

- static DocViewsIterator **ConstructFromImpl** (const DocViewsIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocViewsIterator** (const DocViewsIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocViewsIteratorImpl * pimpl_

  Pointer to internal implementation.

1.116.1 Detailed Description

Basic const-ref iterator for a collection of views CLASS TO BE DEPRECATED.

Definition at line 28 of file doc_views_iterator.h.

1.116.2 Member Data Documentation

pimpl_

```
DocViewsIteratorImpl* se::doc::DocViewsIterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 66 of file doc_views_iterator.h.

1.117 se::doc::DocViewsMutableIterator Class Reference

Mutable-ref iterator for a collection of views.

```
#include <doc_views_iterator.h>
```

Public Member Functions

- **DocViewsMutableIterator** (const DocViewsMutableIterator &other)

  Copy ctor.
- DocViewsMutableIterator & **operator=** (const DocViewsMutableIterator &other)

  Assignment operator.
- ~**DocViewsMutableIterator** ()

  Non-trivial dtor.
- int **GetID** () const

  Returns the view ID.
- const DocView & **GetView** () const

  Returns the view (const ref)
- DocView & **GetMutableView** () const

  Returns the view (mutable ref)
- const DocTagsCollection & **GetTags** () const

  Returns the collection of tags associated with this view.
- const DocView * **GetViewPtr** () const

  Returns the view (const ptr)
- DocView * **GetMutableViewPtr** () const

  Returns the view (mutable ptr)
- const DocTagsCollection * **GetTagsPtr** () const

  Returns the collection of tags associated with this view.
- void **Advance** ()

  Switches the iterator to point on the next view.
- bool **Equals** (const DocViewsMutableIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same view.
- bool **operator==** (const DocViewsMutableIterator &rvalue) const

  Returns true iff this instance and rvalue point to the same view.
- bool **operator!=** (const DocViewsMutableIterator &rvalue) const

  Returns true iff this instance and rvalue point to different views.

Static Public Member Functions

- static DocViewsMutableIterator **ConstructFromImpl** (const DocViewsMutableIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocViewsMutableIterator** (const DocViewsMutableIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocViewsMutableIteratorImpl * pimpl_

  Pointer to internal implementation.

1.117.1 Detailed Description

Mutable-ref iterator for a collection of views.

Definition at line 77 of file doc_views_iterator.h.

1.117.2 Member Data Documentation

pimpl_

```
DocViewsMutableIteratorImpl* se::doc::DocViewsMutableIterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 120 of file doc_views_iterator.h.

1.118 se::doc::DocViewsMutableSliceIterator Class Reference

Mutable-ref iterator for views with a given tag.

```
#include <doc_views_iterator.h>
```

Public Member Functions

- **DocViewsMutableSliceIterator** (const DocViewsMutableSliceIterator &other)
  Copy ctor.
- DocViewsMutableSliceIterator & **operator=** (const DocViewsMutableSliceIterator &other)
  Assignment operator.
- ~**DocViewsMutableSliceIterator** ()
  Non-trivial dtor.
- int **GetID** () const
  Returns the view ID.
- const DocView & **GetView** () const
  Returns the view (const ref)
- DocView & **GetMutableView** () const
  Returns the view (mutable ref)
- const DocTagsCollection & **GetTags** () const
  Returns the collection of tags associated with this view.
- const DocView * **GetViewPtr** () const
  Returns the view (const ptr)
- DocView * **GetMutableViewPtr** () const
  Returns the view (mutable ptr)
- const DocTagsCollection * **GetTagsPtr** () const
  Returns the collection of tags associated with this view.
- void **Advance** ()
  Switches the iterator to point on the next view.
- bool **Finished** () const
  Returns true iff the iterator points to the end of the subset of views with a given tag.

Static Public Member Functions

- static DocViewsMutableSliceIterator **ConstructFromImpl** (const DocViewsMutableSliceIteratorImpl &pimpl)
  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocViewsMutableSliceIterator** (const DocViewsMutableSliceIteratorImpl &pimpl)
  Private ctor from internal implementation.

Private Attributes

- DocViewsMutableSliceIteratorImpl * pimpl_
  Pointer to internal implementation.

1.118.1 Detailed Description

Mutable-ref iterator for views with a given tag.

Definition at line 178 of file doc_views_iterator.h.

1.118.2 Member Data Documentation

pimpl_

```
DocViewsMutableSliceIteratorImpl* se::doc::DocViewsMutableSliceIterator::pimpl_  [private]
```

Pointer to internal implementation.

Definition at line 219 of file doc_views_iterator.h.

1.119 se::doc::DocViewsSliceIterator Class Reference

Const-ref iterator for views with a given tag.

```
#include <doc_views_iterator.h>
```

Public Member Functions

- **DocViewsSliceIterator** (const DocViewsSliceIterator &other)

  Copy ctor.
- DocViewsSliceIterator & **operator=** (const DocViewsSliceIterator &other)

  Assignment operator.
- ~**DocViewsSliceIterator** ()

  Non-trivial dtor.
- int **GetID** () const

  Returns the view ID.
- const DocView & **GetView** () const

  Returns the view (const ref)
- const DocTagsCollection & **GetTags** () const

  Returns the collection of tags associated with this view.
- const DocView * **GetViewPtr** () const

  Returns the view (const ptr)
- const DocTagsCollection * **GetTagsPtr** () const

  Returns the collection of tags associated with this view.
- void **Advance** ()

  Switches the iterator to point on the next view.
- bool **Finished** () const

  Returns true iff the iterator points to the end of the subset of views with a given tag.

Static Public Member Functions

- static DocViewsSliceIterator **ConstructFromImpl** (const DocViewsSliceIteratorImpl &pimpl)

  Factory method - constructs an iterator from its internal implementation.

Private Member Functions

- **DocViewsSliceIterator** (const DocViewsSliceIteratorImpl &pimpl)

  Private ctor from internal implementation.

Private Attributes

- DocViewsSliceIteratorImpl * pimpl_

  Pointer to internal implementation.

1.119.1 Detailed Description

Const-ref iterator for views with a given tag.

Definition at line 131 of file doc_views_iterator.h.

1.119.2 Member Data Documentation

pimpl_

```
DocViewsSliceIteratorImpl* se::doc::DocViewsSliceIterator::pimpl_  [private]
```

Pointer to internal implementation.

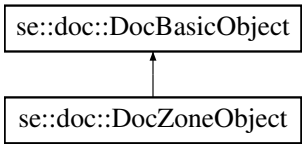
Definition at line 167 of file doc_views_iterator.h.

1.120 se::doc::DocZoneObject Class Reference

The graphical object representing a localized document zone CLASS TO BE DEPRECATED.

```
#include <doc_objects.h>
```

Inheritance diagram for se::doc::DocZoneObject:



Public Member Functions

- virtual ~**DocZoneObject** () override=default
  Default dtor.
- virtual const se::common::Size & **GetSize** () const =0
  Returns the standard pixel size of the zone (const ref)
- virtual se::common::Size & **GetMutableSize** ()=0
  Returns the standard pixel size of the zone (mutable ref)
- virtual const se::common::Size * **GetSizePtr** () const =0
  Returns the standard pixel size of the zone (const ptr)
- virtual se::common::Size * **GetMutableSizePtr** ()=0
  Returns the standard pixel size of the zone (mutable ptr)
- virtual void **SetSize** (const se::common::Size &size)=0
  Sets the standard pixel size of the zone.

Public Member Functions inherited from se::doc::DocBasicObject

- virtual ~**DocBasicObject** ()=default
  Default dtor.
- virtual const char * **ObjectType** () const =0
  Returns the name of the concrete object type.
- virtual const DocBaseObjectInfo & **GetBaseObjectInfo** () const =0
  Returns the general basic object info (const ref)
- virtual DocBaseObjectInfo & **GetMutableBaseObjectInfo** ()=0
  Returns the general basic object info (mutable ref ref)
- virtual const DocBaseObjectInfo * **GetBaseObjectInfoPtr** () const =0
  Returns the general basic object info (const ptr)
- virtual DocBaseObjectInfo * **GetMutableBaseObjectInfoPtr** ()=0
  Returns the general basic object info (mutable ptr)
- virtual void **Serialize** (se::common::Serializer &serializer) const =0
  Serializes the object instance with a given serializer object.

Static Public Member Functions

- static const char * **ObjectTypeStatic** ()

  Returns the object type name.

Static Public Member Functions inherited from se::doc::DocBasicObject

- static const char * **BaseClassNameStatic** ()

  Static class name method, returns 'DocBasicObject'.

1.120.1 Detailed Description

The graphical object representing a localized document zone CLASS TO BE DEPRECATED.

Definition at line 173 of file doc_objects.h.

2 File Documentation

2.1 doc_basic_object.h File Reference

Basic graphical object for Smart Document Engine.

Classes

- class se::doc::DocBaseObjectInfo

  The class representing basic information about a graphical object.
- class se::doc::DocBasicObject

  The class representing a basic graphical object.

2.1.1 Detailed Description

Basic graphical object for Smart Document Engine.

Definition in file doc_basic_object.h.

2.2 doc_basic_object.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_BASIC_OBJECT_H_INCLUDED
00012 #define DOCENGINE_DOC_BASIC_OBJECT_H_INCLUDED
00013
00014 #include <docengine/doc_forward_declarations.h>
00015 #include <secommon/se_common.h>
00016
00017 namespace se { namespace doc {
00018
00019
00023 class SE_DLL_EXPORT DocBaseObjectInfo {
00024 public:
00026   virtual ~DocBaseObjectInfo() = default;
00027
00029   virtual double GetConfidence() const = 0;
00031   virtual bool GetAcceptFlag() const = 0;
00032
00035   virtual const se::common::Polygon& GetGeometryOnPage() const = 0;
00036
00039   virtual const se::common::Polygon* GetGeometryOnPagePtr() const = 0;
00040
00043   virtual const se::common::Polygon& GetGeometryOnScene() const = 0;
00044
00047   virtual const se::common::Polygon* GetGeometryOnScenePtr() const = 0;
00048
00050   virtual int GetAttributesCount() const = 0;
00052   virtual bool HasAttribute(const char* attr_name) const = 0;
00054   virtual const char* GetAttribute(const char* attr_name) const = 0;
00056   virtual se::common::StringsMapIterator AttributesBegin() const = 0;
00058   virtual se::common::StringsMapIterator AttributesEnd() const = 0;
00059
00061   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00062
00063 public:
00067
00070   virtual void SetConfidence(double conf) = 0;
00073   virtual void SetAcceptFlag(bool is_accepted) = 0;
00076   virtual void SetAttribute(const char* attr_name, const char* attr_value) = 0;
00079   virtual void RemoveAttribute(const char* attr_name) = 0;
00080
00084   virtual const se::common::Polygon& GetGeometry() const = 0;
00088   virtual se::common::Polygon& GetMutableGeometry() = 0;
00092   virtual const se::common::Polygon* GetGeometryPtr() const = 0;
00096   virtual se::common::Polygon* GetMutableGeometryPtr() = 0;
00100   virtual void SetGeometry(const se::common::Polygon& geometry) = 0;
00103   virtual int GetViewID() const = 0;
00106   virtual void SetViewID(int view_id) = 0;
00107
00108 };
00109
00110
00114 class SE_DLL_EXPORT DocBasicObject {
00115 public:
00117   static const char* BaseClassNameStatic();
00118
00119 public:
00121   virtual ~DocBasicObject() = default;
00122
00124   virtual const char* ObjectType() const = 0;
00125
00127   virtual const DocBaseObjectInfo& GetBaseObjectInfo() const = 0;
00129   virtual DocBaseObjectInfo& GetMutableBaseObjectInfo() = 0;
00131   virtual const DocBaseObjectInfo* GetBaseObjectInfoPtr() const = 0;
00133   virtual DocBaseObjectInfo* GetMutableBaseObjectInfoPtr() = 0;
00134
00136   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00137 };
00138
00139
00140 } } // namespace se::doc
00141
00142 #endif // DOCENGINE_DOC_BASIC_OBJECT_H_INCLUDED
```

2.3 doc_basic_objects_iterator.h File Reference

Iterators for basic graphical objects.

Classes

- class se::doc::DocBasicObjectsIterator

  Basic const-ref iterator for a collection of basic graphical objects.
- class se::doc::DocTextObjectsIterator
- class se::doc::DocForensicCheckObjectsIterator
- class se::doc::DocImageObjectsIterator
- class se::doc::DocTableObjectsIterator
- class se::doc::DocBarcodeObjectsIterator
- class se::doc::DocCheckboxObjectsIterator
- class se::doc::DocMetaObjectsIterator
- class se::doc::DocBasicObjectsMutableIterator

  Mutable-ref iterator for a collection of basic graphical objects CLASS TO BE DEPRECATED.
- class se::doc::DocBasicObjectsSliceIterator

  Const-ref iterator for a basic objects which have a given tag CLASS TO BE DEPRECATED.
- class se::doc::DocBasicObjectsMutableSliceIterator

  Mutable-ref iterator for a basic objects which have a given tag CLASS TO BE DEPRECATED.
- class se::doc::DocBasicObjectsCrossSliceIterator

  Const-ref iterator for basic objects across multiple collections CLASS TO BE DEPRECATED.
- class se::doc::DocBasicObjectsMutableCrossSliceIterator

  Mutable-ref iterator for basic objects across multiple collections CLASS TO BE DEPRECATED.

2.3.1 Detailed Description

Iterators for basic graphical objects.

Definition in file doc_basic_objects_iterator.h.

2.4 doc_basic_objects_iterator.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_BASIC_OBJECTS_ITERATOR_H_INCLUDED
00012 #define DOCENGINE_DOC_BASIC_OBJECTS_ITERATOR_H_INCLUDED
00013
00014 #include <secommon/se_export_defs.h>
00015 #include <docengine/doc_forward_declarations.h>
00016
00017 namespace se { namespace doc {
00018
00019
00022 class DocBasicObjectsIteratorImpl;
00023
00024 class DocTextObjectsIteratorImpl;
00025 class DocForensicCheckObjectsIteratorImpl;
00026 class DocImageObjectsIteratorImpl;
00027 class DocTableObjectsIteratorImpl;
00028 class DocBarcodeObjectsIteratorImpl;
00029 class DocCheckboxObjectsIteratorImpl;
00030 class DocMetaObjectsIteratorImpl;
00031
00035 class SE_DLL_EXPORT DocBasicObjectsIterator {
00036 private:
00038   DocBasicObjectsIterator(const DocBasicObjectsIteratorImpl& pimpl);
00039
00040 public:
00042   DocBasicObjectsIterator(const DocBasicObjectsIterator& other);
00044   DocBasicObjectsIterator& operator =(const DocBasicObjectsIterator& other);
00046   ~DocBasicObjectsIterator();
```

```
00047
00049   static DocBasicObjectsIterator ConstructFromImpl(
00050       const DocBasicObjectsIteratorImpl& pimpl);
00051
00053   int GetID() const;
00055   const DocBasicObject& GetBasicObject() const;
00057   const DocBasicObject* GetBasicObjectPtr() const;
00059   void Advance();
00060
00062   bool Equals(const DocBasicObjectsIterator& rvalue) const;
00064   bool operator ==(const DocBasicObjectsIterator& rvalue) const;
00066   bool operator !=(const DocBasicObjectsIterator& rvalue) const;
00067
00068 public:
00072
00075   const DocTagsCollection& GetTags() const;
00078   const DocTagsCollection* GetTagsPtr() const;
00079
00080 private:
00082   DocBasicObjectsIteratorImpl* pimpl_;
00083 };
00084
00085 class SE_DLL_EXPORT DocTextObjectsIterator {
00086 private:
00088   DocTextObjectsIterator(const DocTextObjectsIteratorImpl& pimpl);
00089
00090 public:
00092   DocTextObjectsIterator(const DocTextObjectsIterator& other);
00094   DocTextObjectsIterator& operator =(const DocTextObjectsIterator& other);
00096   ~DocTextObjectsIterator();
00097
00099   static DocTextObjectsIterator ConstructFromImpl(
00100       const DocTextObjectsIteratorImpl& pimpl);
00101
00103   const DocTextObject& GetTextObject() const;
00105   const DocTextObject* GetTextObjectPtr() const;
00107   void Advance();
00108
00110   bool Equals(const DocTextObjectsIterator& rvalue) const;
00112   bool operator ==(const DocTextObjectsIterator& rvalue) const;
00114   bool operator !=(const DocTextObjectsIterator& rvalue) const;
00115
00116 private:
00118   DocTextObjectsIteratorImpl* pimpl_;
00119 };
00120
00121
00122 class SE_DLL_EXPORT DocForensicCheckObjectsIterator {
00123 private:
00125   DocForensicCheckObjectsIterator(const DocForensicCheckObjectsIteratorImpl& pimpl);
00126
00127 public:
00129   DocForensicCheckObjectsIterator(const DocForensicCheckObjectsIterator& other);
00131   DocForensicCheckObjectsIterator& operator =(const DocForensicCheckObjectsIterator& other);
00133   ~DocForensicCheckObjectsIterator();
00134
00136   static DocForensicCheckObjectsIterator ConstructFromImpl(
00137       const DocForensicCheckObjectsIteratorImpl& pimpl);
00138
00140   const DocForensicCheckObject& GetForensicCheckObject() const;
00142   const DocForensicCheckObject* GetForensicCheckObjectPtr() const;
00144   void Advance();
00145
00147   bool Equals(const DocForensicCheckObjectsIterator& rvalue) const;
00149   bool operator ==(const DocForensicCheckObjectsIterator& rvalue) const;
00151   bool operator !=(const DocForensicCheckObjectsIterator& rvalue) const;
00152
00153 private:
00155   DocForensicCheckObjectsIteratorImpl* pimpl_;
00156 };
00157
00158 class SE_DLL_EXPORT DocImageObjectsIterator {
00159 private:
00161   DocImageObjectsIterator(const DocImageObjectsIteratorImpl& pimpl);
00162
00163 public:
00165   DocImageObjectsIterator(const DocImageObjectsIterator& other);
00167   DocImageObjectsIterator& operator =(const DocImageObjectsIterator& other);
00169   ~DocImageObjectsIterator();
00170
00172   static DocImageObjectsIterator ConstructFromImpl(
00173       const DocImageObjectsIteratorImpl& pimpl);
00174
00176   const DocImageObject& GetImageObject() const;
00178   const DocImageObject* GetImageObjectPtr() const;
00180   void Advance();
00181
```

```
00183   bool Equals(const DocImageObjectsIterator& rvalue) const;
00185   bool operator ==(const DocImageObjectsIterator& rvalue) const;
00187   bool operator !=(const DocImageObjectsIterator& rvalue) const;
00188
00189 private:
00191   DocImageObjectsIteratorImpl* pimpl_;
00192 };
00193
00194 class SE_DLL_EXPORT DocTableObjectsIterator {
00195 private:
00197   DocTableObjectsIterator(const DocTableObjectsIteratorImpl& pimpl);
00198
00199 public:
00201   DocTableObjectsIterator(const DocTableObjectsIterator& other);
00203   DocTableObjectsIterator& operator =(const DocTableObjectsIterator& other);
00205   ~DocTableObjectsIterator();
00206
00208   static DocTableObjectsIterator ConstructFromImpl(
00209       const DocTableObjectsIteratorImpl& pimpl);
00210
00212   const DocTableObject& GetTableObject() const;
00214   const DocTableObject* GetTableObjectPtr() const;
00216   void Advance();
00217
00219   bool Equals(const DocTableObjectsIterator& rvalue) const;
00221   bool operator ==(const DocTableObjectsIterator& rvalue) const;
00223   bool operator !=(const DocTableObjectsIterator& rvalue) const;
00224
00225 private:
00227   DocTableObjectsIteratorImpl* pimpl_;
00228 };
00229
00230 class SE_DLL_EXPORT DocBarcodeObjectsIterator {
00231 private:
00233   DocBarcodeObjectsIterator(const DocBarcodeObjectsIteratorImpl& pimpl);
00234
00235 public:
00237   DocBarcodeObjectsIterator(const DocBarcodeObjectsIterator& other);
00239   DocBarcodeObjectsIterator& operator =(const DocBarcodeObjectsIterator& other);
00241   ~DocBarcodeObjectsIterator();
00242
00244   static DocBarcodeObjectsIterator ConstructFromImpl(
00245       const DocBarcodeObjectsIteratorImpl& pimpl);
00246
00248   const DocBarcodeObject& GetBarcodeObject() const;
00250   const DocBarcodeObject* GetBarcodeObjectPtr() const;
00252   void Advance();
00253
00255   bool Equals(const DocBarcodeObjectsIterator& rvalue) const;
00257   bool operator ==(const DocBarcodeObjectsIterator& rvalue) const;
00259   bool operator !=(const DocBarcodeObjectsIterator& rvalue) const;
00260
00261 private:
00263   DocBarcodeObjectsIteratorImpl* pimpl_;
00264 };
00265
00266 class SE_DLL_EXPORT DocCheckboxObjectsIterator {
00267 private:
00269   DocCheckboxObjectsIterator(const DocCheckboxObjectsIteratorImpl& pimpl);
00270
00271 public:
00273   DocCheckboxObjectsIterator(const DocCheckboxObjectsIterator& other);
00275   DocCheckboxObjectsIterator& operator =(const DocCheckboxObjectsIterator& other);
00277   ~DocCheckboxObjectsIterator();
00278
00280   static DocCheckboxObjectsIterator ConstructFromImpl(
00281       const DocCheckboxObjectsIteratorImpl& pimpl);
00282
00284   const DocCheckboxObject& GetCheckboxObject() const;
00286   const DocCheckboxObject* GetCheckboxObjectPtr() const;
00288   void Advance();
00289
00291   bool Equals(const DocCheckboxObjectsIterator& rvalue) const;
00293   bool operator ==(const DocCheckboxObjectsIterator& rvalue) const;
00295   bool operator !=(const DocCheckboxObjectsIterator& rvalue) const;
00296
00297 private:
00299   DocCheckboxObjectsIteratorImpl* pimpl_;
00300 };
00301
00302 class SE_DLL_EXPORT DocMetaObjectsIterator {
00303 private:
00305   DocMetaObjectsIterator(const DocMetaObjectsIteratorImpl& pimpl);
00306
00307 public:
00309   DocMetaObjectsIterator(const DocMetaObjectsIterator& other);
00311   DocMetaObjectsIterator& operator =(const DocMetaObjectsIterator& other);
```

```
00313   ~DocMetaObjectsIterator();
00314
00316   static DocMetaObjectsIterator ConstructFromImpl(
00317       const DocMetaObjectsIteratorImpl& pimpl);
00318
00320   const DocMetaObject& GetMetaObject() const;
00322   const DocMetaObject* GetMetaObjectPtr() const;
00324   void Advance();
00325
00327   bool Equals(const DocMetaObjectsIterator& rvalue) const;
00329   bool operator ==(const DocMetaObjectsIterator& rvalue) const;
00331   bool operator !=(const DocMetaObjectsIterator& rvalue) const;
00332
00333 private:
00335   DocMetaObjectsIteratorImpl* pimpl_;
00336 };
00337
00338
00341 class DocBasicObjectsMutableIteratorImpl;
00342
00347 class SE_DLL_EXPORT DocBasicObjectsMutableIterator {
00348 private:
00350   DocBasicObjectsMutableIterator(const DocBasicObjectsMutableIteratorImpl& pimpl);
00351
00352 public:
00354   DocBasicObjectsMutableIterator(const DocBasicObjectsMutableIterator& other);
00356   DocBasicObjectsMutableIterator& operator =(
00357       const DocBasicObjectsMutableIterator& other);
00359   ~DocBasicObjectsMutableIterator();
00360
00362   static DocBasicObjectsMutableIterator ConstructFromImpl(
00363       const DocBasicObjectsMutableIteratorImpl& pimpl);
00364
00366   int GetID() const;
00368   const DocBasicObject& GetBasicObject() const;
00370   DocBasicObject& GetMutableBasicObject() const;
00372   const DocTagsCollection& GetTags() const;
00374   const DocBasicObject* GetBasicObjectPtr() const;
00376   DocBasicObject* GetMutableBasicObjectPtr() const;
00378   const DocTagsCollection* GetTagsPtr() const;
00380   void Advance();
00381
00383   bool Equals(const DocBasicObjectsMutableIterator& rvalue) const;
00385   bool operator ==(const DocBasicObjectsMutableIterator& rvalue) const;
00387   bool operator !=(const DocBasicObjectsMutableIterator& rvalue) const;
00388
00389 private:
00391   DocBasicObjectsMutableIteratorImpl* pimpl_;
00392 };
00393
00394
00397 class DocBasicObjectsSliceIteratorImpl;
00398
00399
00404 class SE_DLL_EXPORT DocBasicObjectsSliceIterator {
00405 private:
00407   DocBasicObjectsSliceIterator(const DocBasicObjectsSliceIteratorImpl& pimpl);
00408
00409 public:
00411   DocBasicObjectsSliceIterator(const DocBasicObjectsSliceIterator& other);
00413   DocBasicObjectsSliceIterator& operator =(
00414       const DocBasicObjectsSliceIterator& other);
00416   ~DocBasicObjectsSliceIterator();
00417
00419   static DocBasicObjectsSliceIterator ConstructFromImpl(
00420       const DocBasicObjectsSliceIteratorImpl& pimpl);
00421
00423   int GetID() const;
00425   const DocBasicObject& GetBasicObject() const;
00427   const DocTagsCollection& GetTags() const;
00429   const DocBasicObject* GetBasicObjectPtr() const;
00431   const DocTagsCollection* GetTagsPtr() const;
00433   void Advance();
00434
00437   bool Finished() const;
00438
00439 private:
00441   DocBasicObjectsSliceIteratorImpl* pimpl_;
00442 };
00443
00444
00447 class DocBasicObjectsMutableSliceIteratorImpl;
00448
00453 class SE_DLL_EXPORT DocBasicObjectsMutableSliceIterator {
00454 private:
00456   DocBasicObjectsMutableSliceIterator(
00457       const DocBasicObjectsMutableSliceIteratorImpl& pimpl);
```

```
00458
00459 public:
00461   DocBasicObjectsMutableSliceIterator(
00462       const DocBasicObjectsMutableSliceIterator& other);
00464   DocBasicObjectsMutableSliceIterator& operator =(
00465       const DocBasicObjectsMutableSliceIterator& other);
00467   ~DocBasicObjectsMutableSliceIterator();
00468
00470   static DocBasicObjectsMutableSliceIterator ConstructFromImpl(
00471       const DocBasicObjectsMutableSliceIteratorImpl& pimpl);
00472
00474   int GetID() const;
00476   const DocBasicObject& GetBasicObject() const;
00478   DocBasicObject& GetMutableBasicObject() const;
00480   const DocTagsCollection& GetTags() const;
00482   const DocBasicObject* GetBasicObjectPtr() const;
00484   DocBasicObject* GetMutableBasicObjectPtr() const;
00486   const DocTagsCollection* GetTagsPtr() const;
00488   void Advance();
00489
00492   bool Finished() const;
00493
00494 private:
00496   DocBasicObjectsMutableSliceIteratorImpl* pimpl_;
00497 };
00498
00499
00502 class DocBasicObjectsCrossSliceIteratorImpl;
00503
00508 class SE_DLL_EXPORT DocBasicObjectsCrossSliceIterator {
00509 private:
00511   DocBasicObjectsCrossSliceIterator(
00512       const DocBasicObjectsCrossSliceIteratorImpl& pimpl);
00513
00514 public:
00516   DocBasicObjectsCrossSliceIterator(
00517       const DocBasicObjectsCrossSliceIterator& other);
00519   DocBasicObjectsCrossSliceIterator& operator =(
00520       const DocBasicObjectsCrossSliceIterator& other);
00522   ~DocBasicObjectsCrossSliceIterator();
00523
00525   static DocBasicObjectsCrossSliceIterator ConstructFromImpl(
00526       const DocBasicObjectsCrossSliceIteratorImpl& pimpl);
00527
00529   int GetCollectionID() const;
00531   int GetObjectID() const;
00533   const DocBasicObject& GetBasicObject() const;
00535   const DocTagsCollection& GetTags() const;
00537   const DocBasicObject* GetBasicObjectPtr() const;
00539   const DocTagsCollection* GetTagsPtr() const;
00541   void Advance();
00542
00544   bool Equals(const DocBasicObjectsCrossSliceIterator& rvalue) const;
00546   bool operator ==(const DocBasicObjectsCrossSliceIterator& rvalue) const;
00548   bool operator !=(const DocBasicObjectsCrossSliceIterator& rvalue) const;
00549
00550 private:
00552   DocBasicObjectsCrossSliceIteratorImpl* pimpl_;
00553 };
00554
00555
00558 class DocBasicObjectsMutableCrossSliceIteratorImpl;
00559
00564 class SE_DLL_EXPORT DocBasicObjectsMutableCrossSliceIterator {
00565 private:
00567   DocBasicObjectsMutableCrossSliceIterator(
00568       const DocBasicObjectsMutableCrossSliceIteratorImpl& pimpl);
00569
00570 public:
00572   DocBasicObjectsMutableCrossSliceIterator(
00573       const DocBasicObjectsMutableCrossSliceIterator& other);
00575   DocBasicObjectsMutableCrossSliceIterator& operator =(
00576       const DocBasicObjectsMutableCrossSliceIterator& other);
00578   ~DocBasicObjectsMutableCrossSliceIterator();
00579
00581   static DocBasicObjectsMutableCrossSliceIterator ConstructFromImpl(
00582       const DocBasicObjectsMutableCrossSliceIteratorImpl& pimpl);
00583
00585   int GetCollectionID() const;
00587   int GetObjectID() const;
00589   const DocBasicObject& GetBasicObject() const;
00591   DocBasicObject& GetMutableBasicObject();
00593   const DocTagsCollection& GetTags() const;
00595   const DocBasicObject* GetBasicObjectPtr() const;
00597   DocBasicObject* GetMutableBasicObjectPtr();
00599   const DocTagsCollection* GetTagsPtr() const;
00601   void Advance();
```

```
00602
00604   bool Equals(const DocBasicObjectsMutableCrossSliceIterator& rvalue) const;
00606   bool operator ==(
00607       const DocBasicObjectsMutableCrossSliceIterator& rvalue) const;
00609   bool operator !=(
00610       const DocBasicObjectsMutableCrossSliceIterator& rvalue) const;
00611
00612 private:
00614   DocBasicObjectsMutableCrossSliceIteratorImpl* pimpl_;
00615 };
00616
00617
00618 } } // namespace se::doc
00619
00620 #endif // DOCENGINE_DOC_BASIC_OBJECTS_ITERATOR_H_INCLUDED
```

2.5 doc_document.h File Reference

Classes of Smart Document Engine document representation.

Classes

- class se::doc::Document

  Class representing a recognized Document.

2.5.1 Detailed Description

Classes of Smart Document Engine document representation.

Definition in file doc_document.h.

2.6 doc_document.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_DOCUMENT_H_INCLUDED
00012 #define DOCENGINE_DOC_DOCUMENT_H_INCLUDED
00013
00014 #include <docengine/doc_fields_iterators.h>
00015 #include <secommon/se_common.h>
00016
00017 namespace se { namespace doc {
00018
00022 class SE_DLL_EXPORT Document {
00023 public:
00025   static const char* BaseClassNameStatic();
00026
00027 public:
00029   virtual ~Document() = default;
00030
00032   virtual int GetTextFieldsCount() const = 0;
00034   virtual bool HasTextField(const char* name) const = 0;
00036   virtual const DocTextField& GetTextField(const char* name) const = 0;
00038   virtual const DocTextField* GetTextFieldPtr(const char* name) const = 0;
00040   virtual DocTextFieldsIterator TextFieldsBegin() const = 0;
00042   virtual DocTextFieldsIterator TextFieldsEnd() const = 0;
00043
00045   virtual int GetImageFieldsCount() const = 0;
00047   virtual bool HasImageField(const char* name) const = 0;
00049   virtual const DocImageField& GetImageField(const char* name) const = 0;
00051   virtual const DocImageField* GetImageFieldPtr(const char* name) const = 0;
00053   virtual DocImageFieldsIterator ImageFieldsBegin() const = 0;
00055   virtual DocImageFieldsIterator ImageFieldsEnd() const = 0;
```

```
00056
00058   virtual int GetCheckboxFieldsCount() const = 0;
00060   virtual bool HasCheckboxField(const char* name) const = 0;
00062   virtual const DocCheckboxField& GetCheckboxField(const char* name) const = 0;
00064   virtual const DocCheckboxField* GetCheckboxFieldPtr(const char* name) const = 0;
00066   virtual DocCheckboxFieldsIterator CheckboxFieldsBegin() const = 0;
00068   virtual DocCheckboxFieldsIterator CheckboxFieldsEnd() const = 0;
00069
00071   virtual int GetForensicFieldsCount() const = 0;
00073   virtual bool HasForensicField(const char* name) const = 0;
00075   virtual const DocForensicField& GetForensicField(const char* name) const = 0;
00077   virtual const DocForensicField* GetForensicFieldPtr(const char* name) const = 0;
00079   virtual DocForensicFieldsIterator ForensicFieldsBegin() const = 0;
00081   virtual DocForensicFieldsIterator ForensicFieldsEnd() const = 0;
00082
00084   virtual int GetForensicCheckFieldsCount() const = 0;
00086   virtual bool HasForensicCheckField(const char* name) const = 0;
00088   virtual const DocForensicCheckField& GetForensicCheckField(const char* name) const = 0;
00090   virtual const DocForensicCheckField* GetForensicCheckFieldPtr(const char* name) const = 0;
00092   virtual DocForensicCheckFieldsIterator ForensicCheckFieldsBegin() const = 0;
00094   virtual DocForensicCheckFieldsIterator ForensicCheckFieldsEnd() const = 0;
00095
00097   virtual int GetTableFieldsCount() const = 0;
00099   virtual bool HasTableField(const char* name) const = 0;
00101   virtual const DocTableField& GetTableField(const char* name) const = 0;
00103   virtual const DocTableField* GetTableFieldPtr(const char* name) const = 0;
00105   virtual DocTableFieldsIterator TableFieldsBegin() const = 0;
00107   virtual DocTableFieldsIterator TableFieldsEnd() const = 0;
00108
00110   virtual int GetBarcodeFieldsCount() const = 0;
00112   virtual bool HasBarcodeField(const char* name) const = 0;
00114   virtual const DocBarcodeField& GetBarcodeField(const char* name) const = 0;
00116   virtual const DocBarcodeField* GetBarcodeFieldPtr(const char* name) const = 0;
00118   virtual DocBarcodeFieldsIterator BarcodeFieldsBegin() const = 0;
00120   virtual DocBarcodeFieldsIterator BarcodeFieldsEnd() const = 0;
00121
00123   virtual int GetAttributesCount() const = 0;
00125   virtual bool HasAttribute(const char* attr_name) const = 0;
00127   virtual const char* GetAttribute(const char* attr_name) const = 0;
00129   virtual void SetAttribute(const char* attr_name, const char* attr_value) = 0;
00131   virtual void RemoveAttribute(const char* attr_name) = 0;
00133   virtual se::common::StringsMapIterator AttributesBegin() const = 0;
00135   virtual se::common::StringsMapIterator AttributesEnd() const = 0;
00136
00138   virtual const char* GetType() const = 0;
00139
00141   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00142
00144   virtual int GetPhysicalDocIdx() const = 0;
00145
00146 public:
00150
00152   virtual DocTextField& GetMutableTextField(const char* name) = 0;
00154   virtual DocTextField* GetMutableTextFieldPtr(const char* name) = 0;
00156   virtual void SetTextField(const char* name, const DocTextField& field) = 0;
00158   virtual void RemoveTextField(const char* name) = 0;
00159
00161   virtual DocImageField& GetMutableImageField(const char* name) = 0;
00163   virtual DocImageField* GetMutableImageFieldPtr(const char* name) = 0;
00165   virtual void SetImageField(const char* name, const DocImageField& field) = 0;
00167   virtual void RemoveImageField(const char* name) = 0;
00168
00170   virtual DocCheckboxField& GetMutableCheckboxField(const char* name) = 0;
00172   virtual DocCheckboxField* GetMutableCheckboxFieldPtr(const char* name) = 0;
00174   virtual void SetCheckboxField(const char* name, const DocCheckboxField& field) = 0;
00176   virtual void RemoveCheckboxField(const char* name) = 0;
00177
00179   virtual DocForensicField& GetMutableForensicField(const char* name) = 0;
00181   virtual DocForensicField* GetMutableForensicFieldPtr(const char* name) = 0;
00183   virtual void SetForensicField(const char* name, const DocForensicField& field) = 0;
00185   virtual void RemoveForensicField(const char* name) = 0;
00186
00188   virtual DocForensicCheckField& GetMutableForensicCheckField(const char* name) = 0;
00190   virtual DocForensicCheckField* GetMutableForensicCheckFieldPtr(const char* name) = 0;
00192   virtual void SetForensicCheckField(const char* name, const DocForensicCheckField& field) = 0;
00194   virtual void RemoveForensicCheckField(const char* name) = 0;
00195
00197   virtual DocTableField& GetMutableTableField(const char* name) = 0;
00199   virtual DocTableField* GetMutableTableFieldPtr(const char* name) = 0;
00201   virtual void SetTableField(const char* name, const DocTableField& field) = 0;
00203   virtual void RemoveTableField(const char* name) = 0;
00204
00206   virtual DocBarcodeField& GetMutableBarcodeField(const char* name) = 0;
00208   virtual DocBarcodeField* GetMutableBarcodeFieldPtr(const char* name) = 0;
00210   virtual void SetBarcodeField(const char* name, const DocBarcodeField& field) = 0;
00212   virtual void RemoveBarcodeField(const char* name) = 0;
00213 };
```

```
00214
00215
00216 } } // namespace se::doc
00217
00218 #endif // DOCENGINE_DOC_DOCUMENT_H_INCLUDED
```

2.7 doc_document_fields_info_iterator.h File Reference

Classes of Smart Document Engine fields iterators.

Classes

- class se::doc::DocDocumentFieldsInfoIterator

  Const-ref iterator for a collection of document fields info.
- class se::doc::DocDocumentTableFieldColumnsInfoInterator

  Const-ref iterator for a collection of columns inside document table field.

2.7.1 Detailed Description

Classes of Smart Document Engine fields iterators.

Definition in file doc_document_fields_info_iterator.h.

2.8 doc_document_fields_info_iterator.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_DOCUMENT_FIELDS_INFO_ITERATOR_H_INCLUDED
00012 #define DOCENGINE_DOC_DOCUMENT_FIELDS_INFO_ITERATOR_H_INCLUDED
00013
00014 #include <secommon/se_export_defs.h>
00015 #include <docengine/doc_forward_declarations.h>
00016
00017 namespace se { namespace doc {
00018
00021 class DocDocumentFieldsInfoIteratorImpl;
00022
00025 class DocDocumentTableFieldColumnsInfoInteratorImpl;
00026
00030 class SE_DLL_EXPORT DocDocumentFieldsInfoIterator {
00031 private:
00033   DocDocumentFieldsInfoIterator(const DocDocumentFieldsInfoIteratorImpl& pimpl);
00034
00035 public:
00037   DocDocumentFieldsInfoIterator(const DocDocumentFieldsInfoIterator& other);
00039   DocDocumentFieldsInfoIterator& operator =(const DocDocumentFieldsInfoIterator& other);
00041   ~DocDocumentFieldsInfoIterator();
00042
00044   static DocDocumentFieldsInfoIterator ConstructFromImpl(
00045       const DocDocumentFieldsInfoIteratorImpl& pimpl);
00046
00048   const char* GetKey() const;
00050   const DocDocumentFieldInfo& GetDocumentFieldInfo() const;
00052   const DocDocumentFieldInfo* GetDocumentFieldInfoPtr() const;
00054   void Advance();
00056   void operator ++();
00057
00059   bool Equals(const DocDocumentFieldsInfoIterator& rvalue) const;
00061   bool operator ==(const DocDocumentFieldsInfoIterator& rvalue) const;
00063   bool operator !=(const DocDocumentFieldsInfoIterator& rvalue) const;
00064
00065 private:
```

```
00067   class DocDocumentFieldsInfoIteratorImpl* pimpl_;
00068 };
00069
00070
00074 class SE_DLL_EXPORT DocDocumentTableFieldColumnsInfoInterator {
00075
00076 private:
00078   DocDocumentTableFieldColumnsInfoInterator(const DocDocumentTableFieldColumnsInfoInteratorImpl&
      pimpl);
00079
00080 public:
00082   DocDocumentTableFieldColumnsInfoInterator(const DocDocumentTableFieldColumnsInfoInterator& other);
00084   DocDocumentTableFieldColumnsInfoInterator& operator =(const
      DocDocumentTableFieldColumnsInfoInterator& other);
00086   ~DocDocumentTableFieldColumnsInfoInterator();
00087
00089   static DocDocumentTableFieldColumnsInfoInterator ConstructFromImpl(
00090     const DocDocumentTableFieldColumnsInfoInteratorImpl& pimpl);
00091
00093   const char* GetKey() const;
00095   const DocDocumentTableFieldColumnInfo& GetDocumentTableFieldColumnInfo() const;
00097   const DocDocumentTableFieldColumnInfo* GetDocumentTableFieldColumnInfoPtr() const;
00099   void Advance();
00101   void operator ++();
00102
00104   bool Equals(const DocDocumentTableFieldColumnsInfoInterator& rvalue) const;
00106   bool operator ==(const DocDocumentTableFieldColumnsInfoInterator& rvalue) const;
00108   bool operator !=(const DocDocumentTableFieldColumnsInfoInterator& rvalue) const;
00109
00110 private:
00112   class DocDocumentTableFieldColumnsInfoInteratorImpl* pimpl_;
00113 };
00114
00115
00116
00117
00118
00119 } } // namespace se::doc
00120
00121 #endif // DOCENGINE_DOC_DOCUMENT_FIELDS_INFO_ITERATOR_H_INCLUDED
```

2.9 doc_document_info.h File Reference

Reference information about document type.

Classes

- class se::doc::DocDocumentInfo

  Reference information about document type.

2.9.1 Detailed Description

Reference information about document type.

Definition in file doc_document_info.h.

2.10 doc_document_info.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_DOCUMENT_INFO_H_INCLUDED
00012 #define DOCENGINE_DOC_DOCUMENT_INFO_H_INCLUDED
00013
```

```
00014 #include <secommon/se_export_defs.h>
00015 #include <docengine/doc_document_fields_info_iterator.h>
00016
00017 namespace se { namespace doc{
00018
00022 class SE_DLL_EXPORT DocDocumentInfo {
00023 public:
00025   virtual ~DocDocumentInfo() = default;
00026
00028   virtual const char* GetDocumentName() const = 0;
00029
00031   virtual const char* GetDocumentNameLocal() const = 0;
00032
00034   virtual const char* GetDocumentShortNameLocal() const = 0;
00035
00037   virtual bool GetDocumentNoFields() const = 0;
00038
00040   virtual DocDocumentFieldsInfoIterator DocumentFieldsInfoBegin() const = 0;
00041
00043   virtual DocDocumentFieldsInfoIterator DocumentFieldsInfoEnd() const = 0;
00044
00046   virtual const DocDocumentFieldInfo& GetDocumentFieldInfo(const char* name) const = 0;
00047
00049   virtual const DocDocumentFieldInfo* GetDocumentFieldInfoPtr(const char* name) const = 0;
00050
00051
00052  public:
00056
00058   virtual bool GetDocumentMultipageInfo() const = 0;
00059 };
00060
00061
00062 }} // namespace se::doc
00063
00064 #endif // DOCENGINE_DOC_DOCUMENT_INFO_H_INCLUDED
```

2.11 doc_documents_iterator.h File Reference

Smart Document Engine documents iterator.

Classes

- class se::doc::DocumentsIterator

  A constant iterator for a collection of Document instances.
- class se::doc::DocumentsMutableIterator

  A mutable iterator for a collection of Document instances CLASS TO BE DEPRECATED.
- class se::doc::DocumentsSliceIterator

  A const iterator for a subset of the collection of Document instances TO BE DEPRECATED.
- class se::doc::DocumentsMutableSliceIterator

  A mutable iterator for a subset of the collection of Document instances TO BE DEPRECATED.

2.11.1 Detailed Description

Smart Document Engine documents iterator.

Definition in file doc_documents_iterator.h.

2.12 doc_documents_iterator.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_DOCUMENTS_ITERATOR_H_INCLUDED
00012 #define DOCENGINE_DOC_DOCUMENTS_ITERATOR_H_INCLUDED
00013
00014 #include <secommon/se_export_defs.h>
00015 #include <docengine/doc_forward_declarations.h>
00016
00017 namespace se { namespace doc {
00018
00021 class DocumentsIteratorImpl;
00022
00026 class SE_DLL_EXPORT DocumentsIterator {
00027 private:
00029   DocumentsIterator(const DocumentsIteratorImpl& pimpl);
00030 public:
00032   DocumentsIterator(const DocumentsIterator& other);
00034   DocumentsIterator& operator =(const DocumentsIterator& other);
00036   ~DocumentsIterator();
00037
00039   static DocumentsIterator ConstructFromImpl(
00040       const DocumentsIteratorImpl& pimpl);
00041
00043   int GetID() const;
00045   const Document& GetDocument() const;
00047   const Document* GetDocumentPtr() const;
00049   void Advance();
00051   void operator ++();
00052
00054   bool Equals(const DocumentsIterator& rvalue) const;
00056   bool operator ==(const DocumentsIterator& rvalue) const;
00058   bool operator !=(const DocumentsIterator& rvalue) const;
00059
00060 public:
00065   const DocTagsCollection& GetTags() const;
00067   const DocTagsCollection* GetTagsPtr() const;
00068
00069 private:
00071   DocumentsIteratorImpl* pimpl_;
00072 };
00073
00076 class DocumentsMutableIteratorImpl;
00077
00082 class SE_DLL_EXPORT DocumentsMutableIterator {
00083 private:
00085   DocumentsMutableIterator(const DocumentsMutableIteratorImpl& pimpl);
00086
00087 public:
00089   DocumentsMutableIterator(const DocumentsMutableIterator& other);
00091   DocumentsMutableIterator& operator =(const DocumentsMutableIterator& other);
00093   ~DocumentsMutableIterator();
00094
00096   static DocumentsMutableIterator ConstructFromImpl(
00097       const DocumentsMutableIteratorImpl& pimpl);
00098
00100   int GetID() const;
00102   const Document& GetDocument() const;
00104   Document& GetMutableDocument() const;
00106   const Document* GetDocumentPtr() const;
00108   Document* GetMutableDocumentPtr() const;
00110   void Advance();
00112   void operator ++();
00113
00115   bool Equals(const DocumentsMutableIterator& rvalue) const;
00117   bool operator ==(const DocumentsMutableIterator& rvalue) const;
00119   bool operator !=(const DocumentsMutableIterator& rvalue) const;
00120
00122   const DocTagsCollection& GetTags() const;
00124   const DocTagsCollection* GetTagsPtr() const;
00125
00126 private:
00128   DocumentsMutableIteratorImpl* pimpl_;
00129 };
00130
00131
00135
00138 class DocumentsSliceIteratorImpl;
00139
00144 class SE_DLL_EXPORT DocumentsSliceIterator {
00145 private:
```

```
00147   DocumentsSliceIterator(const DocumentsSliceIteratorImpl& pimpl);
00148
00149 public:
00151   DocumentsSliceIterator(const DocumentsSliceIterator& other);
00153   DocumentsSliceIterator& operator =(const DocumentsSliceIterator& other);
00155   ~DocumentsSliceIterator();
00156
00158   static DocumentsSliceIterator ConstructFromImpl(
00159       const DocumentsSliceIteratorImpl& pimpl);
00160
00162   int GetID() const;
00164   const Document& GetDocument() const;
00166   const DocTagsCollection& GetTags() const;
00168   const Document* GetDocumentPtr() const;
00170   const DocTagsCollection* GetTagsPtr() const;
00172   void Advance();
00174   void operator ++();
00175
00177   bool Finished() const;
00178
00179 private:
00181   DocumentsSliceIteratorImpl* pimpl_;
00182 };
00183
00184
00187 class DocumentsMutableSliceIteratorImpl;
00188
00194 class SE_DLL_EXPORT DocumentsMutableSliceIterator {
00195 private:
00197   DocumentsMutableSliceIterator(const DocumentsMutableSliceIteratorImpl& pimpl);
00198
00199 public:
00201   DocumentsMutableSliceIterator(const DocumentsMutableSliceIterator& other);
00203   DocumentsMutableSliceIterator& operator =(
00204       const DocumentsMutableSliceIterator& other);
00206   ~DocumentsMutableSliceIterator();
00207
00209   static DocumentsMutableSliceIterator ConstructFromImpl(
00210       const DocumentsMutableSliceIteratorImpl& pimpl);
00211
00213   int GetID() const;
00215   const Document& GetDocument() const;
00217   Document& GetMutableDocument() const;
00219   const DocTagsCollection& GetTags() const;
00221   const Document* GetDocumentPtr() const;
00223   Document* GetMutableDocumentPtr() const;
00225   const DocTagsCollection* GetTagsPtr() const;
00227   void Advance();
00229   void operator ++();
00230
00232   bool Finished() const;
00233
00234 private:
00236   DocumentsMutableSliceIteratorImpl* pimpl_;
00237 };
00238
00239
00240 } } // namespace se::doc
00241
00242 #endif // DOCENGINE_DOC_DOCUMENTS_ITERATOR_H_INCLUDED
```

2.13 doc_engine.h File Reference

Main engine class of Smart Document Engine.

Classes

- class se::doc::DocEngine

  The main DocEngine class containing all configuration and resources of the Smart Document Engine.

2.13.1 Detailed Description

Main engine class of Smart Document Engine.

Definition in file doc_engine.h.

2.14 doc_engine.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_ENGINE_H_INCLUDED
00012 #define DOCENGINE_DOC_ENGINE_H_INCLUDED
00013
00014 #include <secommon/se_common.h>
00015 #include <docengine/doc_forward_declarations.h>
00016
00017 namespace se { namespace doc {
00018
00019
00024 class SE_DLL_EXPORT DocEngine {
00025 public:
00027   virtual ~DocEngine() = default;
00028
00035   virtual DocSessionSettings* CreateSessionSettings() const = 0;
00036
00047   virtual DocSession* SpawnSession(
00048       const DocSessionSettings& settings,
00049       const char*               signature,
00050       DocFeedback*              feedback_reporter = nullptr) const = 0;
00051
00052 public:
00064   static DocEngine* Create(
00065       const char* config_path,
00066       bool        lazy_configuration = true);
00067
00080   static DocEngine* Create(
00081       unsigned char*  config_data,
00082       int             config_data_length,
00083       bool            lazy_configuration = true);
00084
00095   static DocEngine* CreateFromEmbeddedBundle(
00096       bool           lazy_configuration = true);
00097
00102   static const char* GetVersion();
00103
00104
00105  public:
00109
00121   virtual DocSession* SpawnSession(
00122       const DocSessionSettings& settings,
00123       const char*               signature,
00124       DocFeedback*              feedback_reporter,
00125       DocExternalProcessorInterface* external_processor) const = 0;
00126
00127 public:
00131
00139   virtual DocSessionSettings* CreateVideoSessionSettings() const = 0;
00140
00151   virtual DocVideoSession* SpawnVideoSession(
00152     const DocSessionSettings& settings,
00153     const char*               signature,
00154     DocFeedback*              feedback_reporter = nullptr) const = 0;
00155 };
00156
00157
00158 } } // namespace se::doc
00159
00160 #endif // DOCENGINE_DOC_ENGINE_H_INCLUDED
```

2.15 doc_external_processor.h File Reference

Smart Document Engine external processor interface and auxilliary classes.

Classes

- class se::doc::DocProcessingArguments

  The class representing the processing arguments for a custom document processor CLASS TO BE DEPRECATED.
- class se::doc::DocExternalProcessorInterface

  The abstract interface for custom document processor CLASS TO BE DEPRECATED.

2.15.1 Detailed Description

Smart Document Engine external processor interface and auxilliary classes.

Definition in file doc_external_processor.h.

2.16 doc_external_processor.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00012 #ifndef DOCENGINE_DOC_EXTERNAL_PROCESSOR_H_INCLUDED
00013 #define DOCENGINE_DOC_EXTERNAL_PROCESSOR_H_INCLUDED
00014
00015 #include <secommon/se_export_defs.h>
00016 #include <docengine/doc_forward_declarations.h>
00017
00018 namespace se { namespace doc {
00019
00020
00026 class SE_DLL_EXPORT DocProcessingArguments {
00027 public:
00029   virtual ~DocProcessingArguments() = default;
00030
00032   virtual int GetTagArgumentsCount() const = 0;
00034   virtual const char* GetTagArgument(int index) const = 0;
00036   virtual void SetTagArgument(int index, const char* argument) = 0;
00038   virtual void Resize(int size) = 0;
00039 };
00040
00041
00046 class SE_DLL_EXPORT DocExternalProcessorInterface {
00047 public:
00049   virtual ~DocExternalProcessorInterface() = default;
00050
00061   virtual void Process(
00062       DocResult&                   recognition_result,
00063       const DocProcessingSettings&  processing_settings,
00064       const DocProcessingArguments& processing_arguments) = 0;
00065 };
00066
00067
00068 } } // namespace se::doc
00069
00070 #endif // DOCENGINE_DOC_EXTERNAL_PROCESSOR_H_INCLUDED
```

2.17 doc_feedback.h File Reference

Smart Document Engine feedback reporting classes.

Classes

- class se::doc::DocRawFieldFeedback

  The class representing a feedback for one raw field.
- class se::doc::DocRawFieldsFeedbackContainer

  The class representing a feedback container for raw fields.
- class se::doc::DocPageFeedback

  The class representing a feedback for one page.
- class se::doc::DocPagesFeedbackContainer

  The class representing a feedback container for pages.
- class se::doc::DocFeedbackContainer

  The class representing a custom feedback container. Not implemented in the current version of Smart Document Engine CLASS TO BE DEPRECATED.
- class se::doc::DocFeedback

  Abstract interface for receiving Smart Document Engine callbacks. All callbacks must be implemented.

2.17.1 Detailed Description

Smart Document Engine feedback reporting classes.

Definition in file doc_feedback.h.

2.18 doc_feedback.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_FEEDBACK_H_INCLUDED
00012 #define DOCENGINE_DOC_FEEDBACK_H_INCLUDED
00013
00014 #include <secommon/se_common.h>
00015
00016 #include <secommon/se_export_defs.h>
00017 #include <docengine/doc_forward_declarations.h>
00018
00019 namespace se { namespace doc {
00020
00024 class SE_DLL_EXPORT DocRawFieldFeedback {
00025 public:
00027     virtual ~DocRawFieldFeedback() = default;
00028
00030     virtual const char* GetName() const = 0;
00031
00033     virtual bool HasQuadrangle() const = 0;
00034
00036     virtual const se::common::Quadrangle& GetQuadrangle() const = 0;
00037
00039     virtual const char* GetType() const = 0;
00040
00042     virtual const se::common::OcrString GetOcrString() const = 0;
00043 };
00044
00045
00049 class SE_DLL_EXPORT DocRawFieldsFeedbackContainer {
00050 public:
00052     virtual ~DocRawFieldsFeedbackContainer() = default;
00053
00055     virtual int GetRawFieldCount() const = 0;
00056
00058     virtual int GetSourcePageID() const = 0;
00059
00061     virtual const DocRawFieldFeedback& GetRawFieldFeedback(const int idx) const = 0;
00062 };
00063
00067 class SE_DLL_EXPORT DocPageFeedback {
00068 public:
00070     virtual ~DocPageFeedback() = default;
00071
00073     virtual const se::common::Quadrangle& GetQuadrangle() const = 0;
00074
00076     virtual int GetID() const = 0;
00077
00079     virtual const char* GetType() const = 0;
00080
00082     virtual bool IsPageRejected() const = 0;
00083 };
00084
00088 class SE_DLL_EXPORT DocPagesFeedbackContainer {
00089 public:
00091     virtual ~DocPagesFeedbackContainer() = default;
00092
00094     virtual int GetPageCount() const = 0;
00095
00097     virtual const DocPageFeedback& GetPageFeedback(const int idx) const = 0;
00098 };
00099
00105 class SE_DLL_EXPORT DocFeedbackContainer {
00106 public:
00108   virtual ~DocFeedbackContainer() = default;
00110   virtual se::common::StringsMapIterator FeedbackFieldIteratorBegin() const = 0;
00112   virtual se::common::StringsMapIterator FeedbackFieldIteratorEnd() const = 0;
00114   virtual se::common::QuadranglesMapIterator FeedbackQuadIteratorBegin() const = 0;
```

```
00116   virtual se::common::QuadranglesMapIterator FeedbackQuadIteratorEnd() const = 0;
00118   virtual void SetFeedbackField(const char* key, const char* field) = 0;
00120   virtual void SetFeedbackQuad(const char* key, const se::common::Quadrangle& quad) = 0;
00121 };
00122
00123
00128 class SE_DLL_EXPORT DocFeedback {
00129 public:
00131   virtual ~DocFeedback() = default;
00132
00138   virtual void FeedbackReceived(const DocFeedbackContainer& container) = 0;
00139
00142   virtual bool AcceptsPagesLocalizationFeedback() const;
00143
00148   virtual void PagesLocalizationFeedbackReceived(const DocPagesFeedbackContainer& container) const =
      0;
00149
00152   virtual bool AcceptsPagePreprocessingFeedback() const;
00153
00158   virtual void PagePrepocessingFeedbackReceived(const DocPagesFeedbackContainer& container) const = 0;
00159
00162   virtual bool AcceptsRawFieldsLocalizationFeedback() const;
00163
00168   virtual void RawFieldsLocalizationFeedbackReceived(const DocRawFieldsFeedbackContainer& container)
      const = 0;
00169
00172   virtual bool AcceptsRawFieldsRecognitionFeedback() const;
00173
00178   virtual void RawFiedlsRecognitionFeedbackReceived(const DocRawFieldsFeedbackContainer& container)
      const = 0;
00179
00180
00185   virtual void ResultReceived(const DocResult& result_received) = 0;
00186 };
00187
00188
00189 } } // namespace se::doc
00190
00191 #endif // DOCENGINE_DOC_FEEDBACK_H_INCLUDED
```

2.19 doc_fields.h File Reference

Classes of Smart Document Engine fields representation.

Classes

- class se::doc::DocBaseFieldInfo

  The class representing basic document field information.
- class se::doc::DocTextField

  The class representing a text field of a document.
- class se::doc::DocImageField

  The class representing an image field of a document.
- class se::doc::DocCheckboxField

  The class representing a checkbox field of a document.
- class se::doc::DocForensicField

  The class representing a forensic field of a document.
- class se::doc::DocForensicCheckField

  The class representing a forensic check field of a document.
- class se::doc::DocTableField

  The class representing a table field of a document.
- class se::doc::DocBarcodeField

  The class representing a barcode field of a document.

2.19.1 Detailed Description

Classes of Smart Document Engine fields representation.

Definition in file doc_fields.h.

2.20 doc_fields.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_FIELDS_H_INCLUDED
00012 #define DOCENGINE_DOC_FIELDS_H_INCLUDED
00013
00014 #include <secommon/se_common.h>
00015
00016 #include <docengine/doc_forward_declarations.h>
00017 #include <docengine/doc_basic_objects_iterator.h>
00018 #include <docengine/doc_physical_document.h>
00019 #include <docengine/doc_physical_document_iterators.h>
00020
00021
00022 namespace se { namespace doc {
00023
00024
00028 class SE_DLL_EXPORT DocBaseFieldInfo {
00029 public:
00031   virtual ~DocBaseFieldInfo() = default;
00032
00034   virtual const char* GetName() const = 0;
00035
00037   virtual double GetConfidence() const = 0;
00038
00040   virtual bool GetAcceptFlag() const = 0;
00041
00043   virtual bool IsValid() const = 0;
00044
00046   virtual bool IsFictive() const = 0;
00047
00049   virtual int GetAttributesCount() const = 0;
00051   virtual bool HasAttribute(const char* attr_name) const = 0;
00053   virtual const char* GetAttribute(const char* attr_name) const = 0;
00055   virtual se::common::StringsMapIterator AttributesBegin() const = 0;
00057   virtual se::common::StringsMapIterator AttributesEnd() const = 0;
00058
00060   virtual DocTextObjectsCrossPageIterator ConnectedTextObjectsBegin(
00061       const DocPhysicalDocument& phys_doc) const = 0;
00063   virtual DocTextObjectsCrossPageIterator ConnectedTextObjectsEnd(
00064       const DocPhysicalDocument& phys_doc) const = 0;
00065
00067   virtual DocTableObjectsCrossPageIterator ConnectedTableObjectsBegin(
00068       const DocPhysicalDocument& phys_doc) const = 0;
00070   virtual DocTableObjectsCrossPageIterator ConnectedTableObjectsEnd(
00071       const DocPhysicalDocument& phys_doc) const = 0;
00072
00074    virtual DocImageObjectsCrossPageIterator ConnectedImageObjectsBegin(
00075        const DocPhysicalDocument& phys_doc) const = 0;
00077    virtual DocImageObjectsCrossPageIterator ConnectedImageObjectsEnd(
00078        const DocPhysicalDocument& phys_doc) const = 0;
00079
00081    virtual DocCheckboxObjectsCrossPageIterator ConnectedCheckboxObjectsBegin(
00082        const DocPhysicalDocument& phys_doc) const = 0;
00084    virtual DocCheckboxObjectsCrossPageIterator ConnectedCheckboxObjectsEnd(
00085        const DocPhysicalDocument& phys_doc) const = 0;
00086
00088    virtual DocTextObjectsCrossPageIterator ConnectedForensicCheckObjectsBegin(
00089        const DocPhysicalDocument& phys_doc) const = 0;
00091    virtual DocTextObjectsCrossPageIterator ConnectedForensicCheckObjectsEnd(
00092        const DocPhysicalDocument& phys_doc) const = 0;
00093
00095   virtual DocMetaObjectsCrossPageIterator ConnectedForensicObjectsBegin(
00096       const DocPhysicalDocument& phys_doc) const = 0;
00098   virtual DocMetaObjectsCrossPageIterator ConnectedForensicObjectsEnd(
00099       const DocPhysicalDocument& phys_doc) const = 0;
00100
00102   virtual DocBarcodeObjectsCrossPageIterator ConnectedBarcodeObjectsBegin(
```

```
00103       const DocPhysicalDocument& phys_doc) const = 0;
00105   virtual DocBarcodeObjectsCrossPageIterator ConnectedBarcodeObjectsEnd(
00106       const DocPhysicalDocument& phys_doc) const = 0;
00107
00109   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00110
00111 public:
00115
00116
00118   virtual void SetName(const char* name) = 0;
00120   virtual void SetConfidence(double conf) = 0;
00122   virtual void SetAcceptFlag(bool is_accepted) = 0;
00124   virtual void SetAttribute(const char* attr_name, const char* attr_value) = 0;
00126   virtual void RemoveAttribute(const char* attr_name) = 0;
00127
00129   virtual DocBasicObjectsCrossSliceIterator ConnectedBasicObjectsBegin(
00130     const DocGraphicalStructure& graphical) const = 0;
00132   virtual DocBasicObjectsCrossSliceIterator ConnectedBasicObjectsEnd(
00133     const DocGraphicalStructure& graphical) const = 0;
00134
00136   virtual DocBasicObjectsMutableCrossSliceIterator
00137   MutableConnectedBasicObjectsBegin(DocGraphicalStructure& graphical) = 0;
00139   virtual DocBasicObjectsMutableCrossSliceIterator
00140   MutableConnectedBasicObjectsEnd(DocGraphicalStructure& graphical) = 0;
00141
00143   virtual void ConnectBasicObject(int coll_id, int obj_id) = 0;
00144 };
00145
00146
00150 class SE_DLL_EXPORT DocTextField {
00151 public:
00153   virtual ~DocTextField() = default;
00154
00156   virtual const DocBaseFieldInfo& GetBaseFieldInfo() const = 0;
00158   virtual DocBaseFieldInfo& GetMutableBaseFieldInfo() = 0;
00160   virtual const DocBaseFieldInfo* GetBaseFieldInfoPtr() const = 0;
00162   virtual DocBaseFieldInfo* GetMutableBaseFieldInfoPtr() = 0;
00163
00165   virtual const se::common::OcrString& GetOcrString() const = 0;
00167   virtual se::common::OcrString& GetMutableOcrString() = 0;
00169   virtual const se::common::OcrString* GetOcrStringPtr() const = 0;
00171   virtual se::common::OcrString* GetMutableOcrStringPtr() = 0;
00173   virtual void SetOcrString(const se::common::OcrString& ocrstring) = 0;
00174
00176   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00177 };
00178
00179
00183 class SE_DLL_EXPORT DocImageField {
00184 public:
00186   virtual ~DocImageField() = default;
00187
00189   virtual const DocBaseFieldInfo& GetBaseFieldInfo() const = 0;
00191   virtual DocBaseFieldInfo& GetMutableBaseFieldInfo() = 0;
00193   virtual const DocBaseFieldInfo* GetBaseFieldInfoPtr() const = 0;
00195   virtual DocBaseFieldInfo* GetMutableBaseFieldInfoPtr() = 0;
00196
00198   virtual const se::common::Image& GetImage() const = 0;
00200   virtual se::common::Image& GetMutableImage() = 0;
00202   virtual const se::common::Image* GetImagePtr() const = 0;
00204   virtual se::common::Image* GetMutableImagePtr() = 0;
00206   virtual void SetImage(const se::common::Image& image) = 0;
00207
00209   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00210 };
00211
00212
00216 class SE_DLL_EXPORT DocCheckboxField {
00217 public:
00219   virtual ~DocCheckboxField() = default;
00220
00222   virtual const DocBaseFieldInfo& GetBaseFieldInfo() const = 0;
00224   virtual DocBaseFieldInfo& GetMutableBaseFieldInfo() = 0;
00226   virtual const DocBaseFieldInfo* GetBaseFieldInfoPtr() const = 0;
00228   virtual DocBaseFieldInfo* GetMutableBaseFieldInfoPtr() = 0;
00229
00231   virtual bool GetTickStatus() const = 0;
00233   virtual void SetTickStatus(bool tick_status) = 0;
00234
00236   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00237 };
00238
00239
00243 class SE_DLL_EXPORT DocForensicField {
00244 public:
00246   virtual ~DocForensicField() = default;
00247
```

```
00249   virtual const DocBaseFieldInfo& GetBaseFieldInfo() const = 0;
00251   virtual DocBaseFieldInfo& GetMutableBaseFieldInfo() = 0;
00253   virtual const DocBaseFieldInfo* GetBaseFieldInfoPtr() const = 0;
00255   virtual DocBaseFieldInfo* GetMutableBaseFieldInfoPtr() = 0;
00256
00258   virtual const char* GetStatus() const = 0;
00260   virtual void SetStatus(const char* status) = 0;
00261
00263   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00264 };
00265
00269 class SE_DLL_EXPORT DocForensicCheckField {
00270 public:
00272   virtual ~DocForensicCheckField() = default;
00273
00275   virtual const DocBaseFieldInfo& GetBaseFieldInfo() const = 0;
00277   virtual DocBaseFieldInfo& GetMutableBaseFieldInfo() = 0;
00279   virtual const DocBaseFieldInfo* GetBaseFieldInfoPtr() const = 0;
00281   virtual DocBaseFieldInfo* GetMutableBaseFieldInfoPtr() = 0;
00282
00284   virtual const char* GetStatus() const = 0;
00286   virtual void SetStatus(const char* status) = 0;
00287
00289   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00290 };
00291
00292
00296 class SE_DLL_EXPORT DocTableField {
00297 public:
00299   virtual ~DocTableField() = default;
00300
00302   virtual const DocBaseFieldInfo& GetBaseFieldInfo() const = 0;
00304   virtual DocBaseFieldInfo& GetMutableBaseFieldInfo() = 0;
00306   virtual const DocBaseFieldInfo* GetBaseFieldInfoPtr() const = 0;
00308   virtual DocBaseFieldInfo* GetMutableBaseFieldInfoPtr() = 0;
00309
00311   virtual int GetRowsCount() const = 0;
00313   virtual int GetColsCount() const = 0;
00315   virtual const DocTextField& GetCell(int row, int col) const = 0;
00317   virtual DocTextField& GetMutableCell(int row, int col) = 0;
00319   virtual const DocTextField* GetCellPtr(int row, int col) const = 0;
00321   virtual DocTextField* GetMutableCellPtr(int row, int col) = 0;
00323   virtual void SetCell(int row, int col, const DocTextField& text_field) = 0;
00324
00326   virtual bool HasColumnIndexByName(const char* col_name) const = 0;
00328   virtual int GetColumnIndexByName(const char* col_name) const = 0;
00329
00331   virtual void ResizeRows(int rows) = 0;
00333   virtual void ResizeRows(int rows, const DocTextField& filler) = 0;
00335   virtual void ResizeCols(int cols) = 0;
00337   virtual void ResizeCols(int cols, const DocTextField& filler) = 0;
00338
00340   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00341
00343   virtual int GetHeaderRowsCount() const = 0;
00345   virtual int GetHeaderColsCount() const = 0;
00347   virtual const DocTextField& GetHeaderCell(int row, int col) const = 0;
00349   virtual DocTextField& GetHeaderMutableCell(int row, int col) = 0;
00351   virtual const DocTextField* GetHeaderCellPtr(int row, int col) const = 0;
00353   virtual DocTextField* GetHeaderMutableCellPtr(int row, int col) = 0;
00355   virtual void SetHeaderCell(int row, int col, const DocTextField& text_field) = 0;
00356
00358   virtual void ResizeHeaderRows(int rows) = 0;
00360   virtual void ResizeHeaderRows(int rows, const DocTextField& filler) = 0;
00362   virtual void ResizeHeaderCols(int cols) = 0;
00364   virtual void ResizeHeaderCols(int cols, const DocTextField& filler) = 0;
00365
00366
00367 public:
00371
00372
00373
00374
00376   virtual const char* GetColName(int col) const = 0;
00378   virtual void SetColName(int col, const char* col_name) = 0;
00379
00380 };
00381
00382
00386 class SE_DLL_EXPORT DocBarcodeField {
00387 public:
00389   virtual ~DocBarcodeField() = default;
00390
00392   virtual const DocBaseFieldInfo& GetBaseFieldInfo() const = 0;
00394   virtual DocBaseFieldInfo& GetMutableBaseFieldInfo() = 0;
00396   virtual const DocBaseFieldInfo* GetBaseFieldInfoPtr() const = 0;
00398   virtual DocBaseFieldInfo* GetMutableBaseFieldInfoPtr() = 0;
```

```
00399
00401   virtual const se::common::MutableString& GetDecodedString() const = 0;
00403   virtual se::common::MutableString& GetMutableDecodedString() = 0;
00405   virtual const se::common::MutableString* GetDecodedStringPtr() const = 0;
00407   virtual se::common::MutableString* GetMutableDecodedStringPtr() = 0;
00409   virtual void SetDecodedString(const se::common::MutableString& decstring) = 0;
00410
00412   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00413 };
00414
00415
00416 } } // namespace se::doc
00417
00418 #endif // DOCENGINE_DOC_FIELDS_H_INCLUDED
```

2.21 doc_fields_iterators.h File Reference

Classes of Smart Document Engine fields iterators.

Classes

- class se::doc::DocTextFieldsIterator

  Const-ref iterator for a collection of text fields.
- class se::doc::DocImageFieldsIterator

  Const-ref iterator for a collection of image fields.
- class se::doc::DocCheckboxFieldsIterator

  Const-ref iterator for a collection of checkbox fields.
- class se::doc::DocForensicFieldsIterator

  Const-ref iterator for a collection of forensic fields.
- class se::doc::DocForensicCheckFieldsIterator

  Const-ref iterator for a collection of forensic check fields.
- class se::doc::DocTableFieldsIterator

  Const-ref iterator for a collection of table fields.
- class se::doc::DocBarcodeFieldsIterator

  Const-ref iterator for a collection of barcode fields.

2.21.1 Detailed Description

Classes of Smart Document Engine fields iterators.

Definition in file doc_fields_iterators.h.

2.22 doc_fields_iterators.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_FIELDS_ITERATORS_H_INCLUDED
00012 #define DOCENGINE_DOC_FIELDS_ITERATORS_H_INCLUDED
00013
00014 #include <secommon/se_export_defs.h>
00015 #include <docengine/doc_forward_declarations.h>
00016
00017 namespace se { namespace doc {
00018
```

```
00021 class DocTextFieldsIteratorImpl;
00022
00026 class SE_DLL_EXPORT DocTextFieldsIterator {
00027 private:
00029   DocTextFieldsIterator(const DocTextFieldsIteratorImpl& pimpl);
00030
00031 public:
00033   DocTextFieldsIterator(const DocTextFieldsIterator& other);
00035   DocTextFieldsIterator& operator =(const DocTextFieldsIterator& other);
00037   ~DocTextFieldsIterator();
00038
00040   static DocTextFieldsIterator ConstructFromImpl(
00041       const DocTextFieldsIteratorImpl& pimpl);
00042
00044   const char* GetKey() const;
00046   const DocTextField& GetField() const;
00048   const DocTextField* GetFieldPtr() const;
00050   void Advance();
00052   void operator ++();
00053
00055   bool Equals(const DocTextFieldsIterator& rvalue) const;
00057   bool operator ==(const DocTextFieldsIterator& rvalue) const;
00059   bool operator !=(const DocTextFieldsIterator& rvalue) const;
00060
00061 private:
00063   class DocTextFieldsIteratorImpl* pimpl_;
00064 };
00065
00066
00069 class DocImageFieldsIteratorImpl;
00070
00074 class SE_DLL_EXPORT DocImageFieldsIterator {
00075 private:
00077   DocImageFieldsIterator(const DocImageFieldsIteratorImpl& pimpl);
00078
00079 public:
00081   DocImageFieldsIterator(const DocImageFieldsIterator& other);
00083   DocImageFieldsIterator& operator =(const DocImageFieldsIterator& other);
00085   ~DocImageFieldsIterator();
00086
00088   static DocImageFieldsIterator ConstructFromImpl(
00089       const DocImageFieldsIteratorImpl& pimpl);
00090
00092   const char* GetKey() const;
00094   const DocImageField& GetField() const;
00096   const DocImageField* GetFieldPtr() const;
00098   void Advance();
00100   void operator ++();
00101
00103   bool Equals(const DocImageFieldsIterator& rvalue) const;
00105   bool operator ==(const DocImageFieldsIterator& rvalue) const;
00107   bool operator !=(const DocImageFieldsIterator& rvalue) const;
00108
00109 private:
00111   class DocImageFieldsIteratorImpl* pimpl_;
00112 };
00113
00114
00117 class DocCheckboxFieldsIteratorImpl;
00118
00122 class SE_DLL_EXPORT DocCheckboxFieldsIterator {
00123 private:
00125   DocCheckboxFieldsIterator(const DocCheckboxFieldsIteratorImpl& pimpl);
00126
00127 public:
00129   DocCheckboxFieldsIterator(const DocCheckboxFieldsIterator& other);
00131   DocCheckboxFieldsIterator& operator =(const DocCheckboxFieldsIterator& other);
00133   ~DocCheckboxFieldsIterator();
00134
00136   static DocCheckboxFieldsIterator ConstructFromImpl(
00137       const DocCheckboxFieldsIteratorImpl& pimpl);
00138
00140   const char* GetKey() const;
00142   const DocCheckboxField& GetField() const;
00144   const DocCheckboxField* GetFieldPtr() const;
00146   void Advance();
00148   void operator ++();
00149
00151   bool Equals(const DocCheckboxFieldsIterator& rvalue) const;
00153   bool operator ==(const DocCheckboxFieldsIterator& rvalue) const;
00155   bool operator !=(const DocCheckboxFieldsIterator& rvalue) const;
00156
00157 private:
00159   class DocCheckboxFieldsIteratorImpl* pimpl_;
00160 };
00161
00162
```

```
00165 class DocForensicFieldsIteratorImpl;
00166
00170 class SE_DLL_EXPORT DocForensicFieldsIterator {
00171 private:
00173   DocForensicFieldsIterator(const DocForensicFieldsIteratorImpl& pimpl);
00174
00175 public:
00177   DocForensicFieldsIterator(const DocForensicFieldsIterator& other);
00179   DocForensicFieldsIterator& operator =(const DocForensicFieldsIterator& other);
00181   ~DocForensicFieldsIterator();
00182
00184   static DocForensicFieldsIterator ConstructFromImpl(
00185       const DocForensicFieldsIteratorImpl& pimpl);
00186
00188   const char* GetKey() const;
00190   const DocForensicField& GetField() const;
00192   const DocForensicField* GetFieldPtr() const;
00194   void Advance();
00196   void operator ++();
00197
00199   bool Equals(const DocForensicFieldsIterator& rvalue) const;
00201   bool operator ==(const DocForensicFieldsIterator& rvalue) const;
00203   bool operator !=(const DocForensicFieldsIterator& rvalue) const;
00204
00205 private:
00207   class DocForensicFieldsIteratorImpl* pimpl_;
00208 };
00209
00212 class DocForensicCheckFieldsIteratorImpl;
00213
00217 class SE_DLL_EXPORT DocForensicCheckFieldsIterator {
00218 private:
00220   DocForensicCheckFieldsIterator(const DocForensicCheckFieldsIteratorImpl& pimpl);
00221
00222 public:
00224   DocForensicCheckFieldsIterator(const DocForensicCheckFieldsIterator& other);
00226   DocForensicCheckFieldsIterator& operator =(const DocForensicCheckFieldsIterator& other);
00228   ~DocForensicCheckFieldsIterator();
00229
00231   static DocForensicCheckFieldsIterator ConstructFromImpl(
00232       const DocForensicCheckFieldsIteratorImpl& pimpl);
00233
00235   const char* GetKey() const;
00237   const DocForensicCheckField& GetField() const;
00239   const DocForensicCheckField* GetFieldPtr() const;
00241   void Advance();
00243   void operator ++();
00244
00246   bool Equals(const DocForensicCheckFieldsIterator& rvalue) const;
00248   bool operator ==(const DocForensicCheckFieldsIterator& rvalue) const;
00250   bool operator !=(const DocForensicCheckFieldsIterator& rvalue) const;
00251
00252 private:
00254   class DocForensicCheckFieldsIteratorImpl* pimpl_;
00255 };
00256
00257
00260 class DocTableFieldsIteratorImpl;
00261
00265 class SE_DLL_EXPORT DocTableFieldsIterator {
00266 private:
00268   DocTableFieldsIterator(const DocTableFieldsIteratorImpl& pimpl);
00269
00270 public:
00272   DocTableFieldsIterator(const DocTableFieldsIterator& other);
00274   DocTableFieldsIterator& operator =(const DocTableFieldsIterator& other);
00276   ~DocTableFieldsIterator();
00277
00279   static DocTableFieldsIterator ConstructFromImpl(
00280       const DocTableFieldsIteratorImpl& pimpl);
00281
00283   const char* GetKey() const;
00285   const DocTableField& GetField() const;
00287   const DocTableField* GetFieldPtr() const;
00289   void Advance();
00291   void operator ++();
00292
00294   bool Equals(const DocTableFieldsIterator& rvalue) const;
00296   bool operator ==(const DocTableFieldsIterator& rvalue) const;
00298   bool operator !=(const DocTableFieldsIterator& rvalue) const;
00299
00300 private:
00302   class DocTableFieldsIteratorImpl* pimpl_;
00303 };
00304
00305
00308 class DocBarcodeFieldsIteratorImpl;
```

```
00309
00313 class SE_DLL_EXPORT DocBarcodeFieldsIterator {
00314 private:
00316   DocBarcodeFieldsIterator(const DocBarcodeFieldsIteratorImpl& pimpl);
00317
00318 public:
00320   DocBarcodeFieldsIterator(const DocBarcodeFieldsIterator& other);
00322   DocBarcodeFieldsIterator& operator =(const DocBarcodeFieldsIterator& other);
00324   ~DocBarcodeFieldsIterator();
00325
00327   static DocBarcodeFieldsIterator ConstructFromImpl(
00328       const DocBarcodeFieldsIteratorImpl& pimpl);
00329
00331   const char* GetKey() const;
00333   const DocBarcodeField& GetField() const;
00335   const DocBarcodeField* GetFieldPtr() const;
00337   void Advance();
00339   void operator ++();
00340
00342   bool Equals(const DocBarcodeFieldsIterator& rvalue) const;
00344   bool operator ==(const DocBarcodeFieldsIterator& rvalue) const;
00346   bool operator !=(const DocBarcodeFieldsIterator& rvalue) const;
00347
00348 private:
00350   class DocBarcodeFieldsIteratorImpl* pimpl_;
00351 };
00352
00353 } } // namespace se::doc
00354
00355 #endif // DOCENGINE_DOC_FIELDS_ITERATORS_H_INCLUDED
```

2.23 doc_forward_declarations.h File Reference

Forward declarations for Smart Document Engine classes.

Variables

- class SE_DLL_EXPORT se::doc::DocTagsCollection
- class SE_DLL_EXPORT se::doc::DocView
- class SE_DLL_EXPORT se::doc::DocViewsCollection
- class SE_DLL_EXPORT se::doc::DocBaseObjectInfo
- class SE_DLL_EXPORT se::doc::DocBasicObject
- class SE_DLL_EXPORT se::doc::DocObjectsCollection
- class SE_DLL_EXPORT se::doc::DocGraphicalStructure
- class SE_DLL_EXPORT se::doc::DocTemplateObject
- class SE_DLL_EXPORT se::doc::DocTextObject
- class SE_DLL_EXPORT se::doc::DocForensicCheckObject
- class SE_DLL_EXPORT se::doc::DocImageObject
- class SE_DLL_EXPORT se::doc::DocTableObject
- class SE_DLL_EXPORT se::doc::DocMultiStringTextObjectImpl
- class SE_DLL_EXPORT se::doc::DocZoneObject
- class SE_DLL_EXPORT se::doc::DocCheckboxObject
- class SE_DLL_EXPORT se::doc::DocLineObject
- class SE_DLL_EXPORT se::doc::DocMetaObject
- class SE_DLL_EXPORT se::doc::DocBarcodeObject
- class SE_DLL_EXPORT se::doc::DocMarkObject
- class SE_DLL_EXPORT se::doc::DocTextField
- class SE_DLL_EXPORT se::doc::DocImageField
- class SE_DLL_EXPORT se::doc::DocCheckboxField
- class SE_DLL_EXPORT se::doc::DocForensicField
- class SE_DLL_EXPORT se::doc::DocForensicCheckField
- class SE_DLL_EXPORT se::doc::DocTableField
- class SE_DLL_EXPORT se::doc::DocBarcodeField

- class SE_DLL_EXPORT se::doc::Document
- class SE_DLL_EXPORT se::doc::DocResult
- class SE_DLL_EXPORT se::doc::DocSessionSettings
- class SE_DLL_EXPORT se::doc::DocSession
- class SE_DLL_EXPORT se::doc::DocVideoSession
- class SE_DLL_EXPORT se::doc::DocProcessingSettings
- class SE_DLL_EXPORT se::doc::DocFeedback
- class SE_DLL_EXPORT se::doc::DocProcessingArguments
- class SE_DLL_EXPORT se::doc::DocExternalProcessorInterface
- class SE_DLL_EXPORT se::doc::DocDocumentFieldInfo
- class SE_DLL_EXPORT se::doc::DocDocumentTableFieldColumnInfo

2.23.1 Detailed Description

Forward declarations for Smart Document Engine classes.

Definition in file doc_forward_declarations.h.

2.23.2 Variable Documentation

DocTagsCollection

```
class SE_DLL_EXPORT se::doc::DocTagsCollection
```

Definition at line 18 of file doc_forward_declarations.h.

DocView

```
class SE_DLL_EXPORT se::doc::DocView
```

Definition at line 20 of file doc_forward_declarations.h.

DocViewsCollection

```
class SE_DLL_EXPORT se::doc::DocViewsCollection
```

Definition at line 21 of file doc_forward_declarations.h.

DocBaseObjectInfo

```
class SE_DLL_EXPORT se::doc::DocBaseObjectInfo
```

Definition at line 22 of file doc_forward_declarations.h.

DocBasicObject

```
class SE_DLL_EXPORT se::doc::DocBasicObject
```

Definition at line 23 of file doc_forward_declarations.h.

DocObjectsCollection

```
class SE_DLL_EXPORT se::doc::DocObjectsCollection
```

Definition at line 24 of file doc_forward_declarations.h.

DocGraphicalStructure

```
class SE_DLL_EXPORT se::doc::DocGraphicalStructure
```

Definition at line 25 of file doc_forward_declarations.h.

DocTemplateObject

```
class SE_DLL_EXPORT se::doc::DocTemplateObject
```

Definition at line 27 of file doc_forward_declarations.h.

DocTextObject

```
class SE_DLL_EXPORT se::doc::DocTextObject
```

Definition at line 28 of file doc_forward_declarations.h.

DocForensicCheckObject

```
class SE_DLL_EXPORT se::doc::DocForensicCheckObject
```

Definition at line 29 of file doc_forward_declarations.h.

DocImageObject

```
class SE_DLL_EXPORT se::doc::DocImageObject
```

Definition at line 30 of file doc_forward_declarations.h.

DocTableObject

```
class SE_DLL_EXPORT se::doc::DocTableObject
```

Definition at line 31 of file doc_forward_declarations.h.

DocMultiStringTextObjectImpl

```
class SE_DLL_EXPORT se::doc::DocMultiStringTextObjectImpl
```

Definition at line 32 of file doc_forward_declarations.h.

DocZoneObject

```
class SE_DLL_EXPORT se::doc::DocZoneObject
```

Definition at line 33 of file doc_forward_declarations.h.

DocCheckboxObject

```
class SE_DLL_EXPORT se::doc::DocCheckboxObject
```

Definition at line 34 of file doc_forward_declarations.h.

DocLineObject

```
class SE_DLL_EXPORT se::doc::DocLineObject
```

Definition at line 35 of file doc_forward_declarations.h.

DocMetaObject

```
class SE_DLL_EXPORT se::doc::DocMetaObject
```

Definition at line 37 of file doc_forward_declarations.h.

DocBarcodeObject

```
class SE_DLL_EXPORT se::doc::DocBarcodeObject
```

Definition at line 38 of file doc_forward_declarations.h.

DocMarkObject

```
class SE_DLL_EXPORT se::doc::DocMarkObject
```

Definition at line 39 of file doc_forward_declarations.h.

DocTextField

```
class SE_DLL_EXPORT se::doc::DocTextField
```

Definition at line 41 of file doc_forward_declarations.h.

DocImageField

```
class SE_DLL_EXPORT se::doc::DocImageField
```

Definition at line 42 of file doc_forward_declarations.h.

DocCheckboxField

```
class SE_DLL_EXPORT se::doc::DocCheckboxField
```

Definition at line 43 of file doc_forward_declarations.h.

DocForensicField

```
class SE_DLL_EXPORT se::doc::DocForensicField
```

Definition at line 44 of file doc_forward_declarations.h.

DocForensicCheckField

```
class SE_DLL_EXPORT se::doc::DocForensicCheckField
```

Definition at line 45 of file doc_forward_declarations.h.

DocTableField

```
class SE_DLL_EXPORT se::doc::DocTableField
```

Definition at line 46 of file doc_forward_declarations.h.

DocBarcodeField

```
class SE_DLL_EXPORT se::doc::DocBarcodeField
```

Definition at line 47 of file doc_forward_declarations.h.

Document

```
class SE_DLL_EXPORT se::doc::Document
```

Definition at line 48 of file doc_forward_declarations.h.

DocResult

```
class SE_DLL_EXPORT se::doc::DocResult
```

Definition at line 50 of file doc_forward_declarations.h.

DocSessionSettings

```
class SE_DLL_EXPORT se::doc::DocSessionSettings
```

Definition at line 52 of file doc_forward_declarations.h.

DocSession

```
class SE_DLL_EXPORT se::doc::DocSession
```

Definition at line 53 of file doc_forward_declarations.h.

DocVideoSession

```
class SE_DLL_EXPORT se::doc::DocVideoSession
```

Definition at line 54 of file doc_forward_declarations.h.

DocProcessingSettings

```
class SE_DLL_EXPORT se::doc::DocProcessingSettings
```

Definition at line 55 of file doc_forward_declarations.h.

DocFeedback

```
class SE_DLL_EXPORT se::doc::DocFeedback
```

Definition at line 56 of file doc_forward_declarations.h.

DocProcessingArguments

```
class SE_DLL_EXPORT se::doc::DocProcessingArguments
```

Definition at line 57 of file doc_forward_declarations.h.

DocExternalProcessorInterface

```
class SE_DLL_EXPORT se::doc::DocExternalProcessorInterface
```

Definition at line 58 of file doc_forward_declarations.h.

DocDocumentFieldInfo

```
class SE_DLL_EXPORT se::doc::DocDocumentFieldInfo
```

Definition at line 60 of file doc_forward_declarations.h.

DocDocumentTableFieldColumnInfo

```
class SE_DLL_EXPORT se::doc::DocDocumentTableFieldColumnInfo
```

Definition at line 61 of file doc_forward_declarations.h.

2.24 doc_forward_declarations.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_FORWARD_DECLARATIONS_H_INCLUDED
00012 #define DOCENGINE_DOC_FORWARD_DECLARATIONS_H_INCLUDED
00013
00014 #include <secommon/se_export_defs.h>
00015
00016 namespace se { namespace doc {
00017
00018 class SE_DLL_EXPORT DocTagsCollection;
00019
00020 class SE_DLL_EXPORT DocView;
00021 class SE_DLL_EXPORT DocViewsCollection;
00022 class SE_DLL_EXPORT DocBaseObjectInfo;
00023 class SE_DLL_EXPORT DocBasicObject;
00024 class SE_DLL_EXPORT DocObjectsCollection;
00025 class SE_DLL_EXPORT DocGraphicalStructure;
00026
00027 class SE_DLL_EXPORT DocTemplateObject;
00028 class SE_DLL_EXPORT DocTextObject;
00029 class SE_DLL_EXPORT DocForensicCheckObject;
00030 class SE_DLL_EXPORT DocImageObject;
00031 class SE_DLL_EXPORT DocTableObject;
00032 class SE_DLL_EXPORT DocMultiStringTextObjectImpl;
00033 class SE_DLL_EXPORT DocZoneObject;
00034 class SE_DLL_EXPORT DocCheckboxObject;
00035 class SE_DLL_EXPORT DocLineObject;
00036 class SE_DLL_EXPORT DocTableObject;
00037 class SE_DLL_EXPORT DocMetaObject;
00038 class SE_DLL_EXPORT DocBarcodeObject;
00039 class SE_DLL_EXPORT DocMarkObject;
00040
00041 class SE_DLL_EXPORT DocTextField;
00042 class SE_DLL_EXPORT DocImageField;
00043 class SE_DLL_EXPORT DocCheckboxField;
00044 class SE_DLL_EXPORT DocForensicField;
00045 class SE_DLL_EXPORT DocForensicCheckField;
00046 class SE_DLL_EXPORT DocTableField;
00047 class SE_DLL_EXPORT DocBarcodeField;
00048 class SE_DLL_EXPORT Document;
00049
00050 class SE_DLL_EXPORT DocResult;
00051
00052 class SE_DLL_EXPORT DocSessionSettings;
00053 class SE_DLL_EXPORT DocSession;
00054 class SE_DLL_EXPORT DocVideoSession;
00055 class SE_DLL_EXPORT DocProcessingSettings;
00056 class SE_DLL_EXPORT DocFeedback;
00057 class SE_DLL_EXPORT DocProcessingArguments;
00058 class SE_DLL_EXPORT DocExternalProcessorInterface;
00059
00060 class SE_DLL_EXPORT DocDocumentFieldInfo;
00061 class SE_DLL_EXPORT DocDocumentTableFieldColumnInfo;
00062
00063 } } // namespace se::doc
00064
00065 #endif // DOCENGINE_DOC_FORWARD_DECLARATIONS_H_INCLUDED
```

2.25 doc_graphical_structure.h File Reference

Classes of Smart Document Engine graphical result structure.

Classes

- class se::doc::DocGraphicalStructure

  The class represting a graphical structure - a result of graphical document processing and graphical objects extraction CLASS TO BE DEPRECATED.

2.25.1 Detailed Description

Classes of Smart Document Engine graphical result structure.

Definition in file doc_graphical_structure.h.

2.26 doc_graphical_structure.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_GRAPHICAL_STRUCTURE_H_INCLUDED
00012 #define DOCENGINE_DOC_GRAPHICAL_STRUCTURE_H_INCLUDED
00013
00014 #include <secommon/se_common.h>
00015 #include <docengine/doc_forward_declarations.h>
00016 #include <docengine/doc_objects_collections_iterator.h>
00017
00018 namespace se { namespace doc {
00019
00020
00026 class SE_DLL_EXPORT DocGraphicalStructure {
00027 public:
00029   virtual ~DocGraphicalStructure() = default;
00030
00032   virtual int GetCollectionsCount() const = 0;
00034   virtual bool HasCollection(int c_id) const = 0;
00036   virtual const DocObjectsCollection& GetCollection(int c_id) const = 0;
00038   virtual DocObjectsCollection& GetMutableCollection(int c_id) = 0;
00040   virtual const DocTagsCollection& GetCollectionTags(int c_id) const = 0;
00042   virtual const DocObjectsCollection* GetCollectionPtr(int c_id) const = 0;
00044   virtual DocObjectsCollection* GetMutableCollectionPtr(int c_id) = 0;
00046   virtual const DocTagsCollection* GetCollectionTagsPtr(int c_id) const = 0;
00048   virtual DocObjectsCollectionsMutableIterator AddCollection(
00049       const DocObjectsCollection& collection) = 0;
00051   virtual DocObjectsCollectionsMutableIterator AddCollection(
00052       const DocObjectsCollection& collection,
00053       const DocTagsCollection& tags) = 0;
00055   virtual void SetCollection(
00056       int c_id, const DocObjectsCollection& collection) = 0;
00058   virtual void RemoveCollection(int c_id) = 0;
00059
00061   virtual DocObjectsCollectionsIterator ObjectsCollectionsBegin() const = 0;
00063   virtual DocObjectsCollectionsIterator ObjectsCollectionsEnd() const = 0;
00064
00066   virtual DocObjectsCollectionsMutableIterator
00067   MutableObjectsCollectionsBegin() = 0;
00069   virtual DocObjectsCollectionsMutableIterator
00070   MutableObjectsCollectionsEnd() = 0;
00071
00074   virtual DocObjectsCollectionsSliceIterator ObjectsCollectionsSlice(
00075       const char* tag) const = 0;
00076
00079   virtual DocObjectsCollectionsMutableSliceIterator MutableObjectsCollectionsSlice(
00080       const char* tag) = 0;
00081
00083   virtual const DocViewsCollection& GetViewsCollection() const = 0;
00085   virtual DocViewsCollection& GetMutableViewsCollection() = 0;
00087   virtual const DocViewsCollection* GetViewsCollectionPtr() const = 0;
00089   virtual DocViewsCollection* GetMutableViewsCollectionPtr() = 0;
00090
00092   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00093 };
00094
00095
00096 } } // namespace se::doc
00097
00098 #endif // DOCENGINE_DOC_GRAPHICAL_STRUCTURE_H_INCLUDED
```

2.27 doc_objects.h File Reference

Types of graphical structure objects of Smart Document Engine.

Classes

- class se::doc::DocTextLineObject

  The graphical object representing a text line.
- class se::doc::DocTextObject

  The graphical object representing a text.
- class se::doc::DocForensicCheckObject

  The graphical object representing a forensic check.
- class se::doc::DocCheckboxObject

  The graphical object representing a checkbox.
- class se::doc::DocTemplateObject

  The graphical object representing a fixed subform template.
- class se::doc::DocLineObject

  The graphical object representing a straight line segment.
- class se::doc::DocZoneObject

  The graphical object representing a localized document zone CLASS TO BE DEPRECATED.
- class se::doc::DocMultiStringTextObject

  The graphical object representing a text object with multiple lines CLASS TO BE DEPRECATED.
- class se::doc::DocMetaObject

  The graphical object representing a meta object.
- class se::doc::DocTableObject

  The graphical object representing a table.
- class se::doc::DocImageObject

  The graphical object representing an image region of a document.
- class se::doc::DocBarcodeObject

  The graphical object representing a barcode.
- class se::doc::DocMarkObject

  The graphical object representing a remark or correction on a document.

2.27.1 Detailed Description

Types of graphical structure objects of Smart Document Engine.

Definition in file doc_objects.h.

2.28 doc_objects.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_OBJECTS_H_INCLUDED
00012 #define DOCENGINE_DOC_OBJECTS_H_INCLUDED
00013
00014 #include <secommon/se_common.h>
00015 #include <docengine/doc_forward_declarations.h>
00016 #include <docengine/doc_basic_object.h>
00017
00018 namespace se { namespace doc {
00019
00023 class SE_DLL_EXPORT DocTextLineObject : public DocBasicObject {
00024 public:
00026   virtual ~DocTextLineObject() override = default;
00027
00029   static const char* ObjectTypeStatic();
00030
00032   virtual const se::common::OcrString& GetOcrString() const = 0;
00034   virtual const se::common::OcrString* GetOcrStringPtr() const = 0;
00035
00036   virtual const se::common::QuadranglesVectorIterator CellQuadranglesOnSceneBegin() const = 0;
00037
00038   virtual const se::common::QuadranglesVectorIterator CellQuadranglesOnSceneEnd() const = 0;
00039
00040   virtual const se::common::QuadranglesVectorIterator CellQuadranglesOnPageBegin() const = 0;
00041
00042   virtual const se::common::QuadranglesVectorIterator CellQuadranglesOnPageEnd() const = 0;
00043
00044   virtual const se::common::Quadrangle& GetCellQuadOnScene(int idx) const = 0;
00045
00046   virtual const se::common::Quadrangle& GetCellQuadOnPage(int idx) const = 0;
00047 };
00048
00049
00053 class SE_DLL_EXPORT DocTextObject : public DocBasicObject {
00054 public:
00056   virtual ~DocTextObject() override = default;
00057
00059   static const char* ObjectTypeStatic();
00060
00062   virtual const se::common::OcrString& GetOcrString() const = 0;
00064   virtual const se::common::OcrString* GetOcrStringPtr() const = 0;
00066   virtual int GetTextLineObjectsCount() const = 0;
00068   virtual const DocTextLineObject& GetTextLineObject(int index) const = 0;
00070   virtual const DocTextLineObject* GetTextLineObjectPtr(int index) const = 0;
00071
00072 public:
00076
00078   virtual se::common::OcrString& GetMutableOcrString() = 0;
00080   virtual se::common::OcrString* GetMutableOcrStringPtr() = 0;
00082   virtual void SetOcrString(const se::common::OcrString& ocrstring) = 0;
00083 };
00084
00088 class SE_DLL_EXPORT DocForensicCheckObject : public DocBasicObject {
00089 public:
00091   virtual ~DocForensicCheckObject() override = default;
00092
00094   static const char* ObjectTypeStatic();
00095
00097   virtual const se::common::OcrString& GetOcrString() const = 0;
00099   virtual const se::common::OcrString* GetOcrStringPtr() const = 0;
00100
00101
00102 public:
00106
00108   virtual se::common::OcrString& GetMutableOcrString() = 0;
00110   virtual se::common::OcrString* GetMutableOcrStringPtr() = 0;
00111 };
00112
00116 class SE_DLL_EXPORT DocCheckboxObject : public DocBasicObject {
00117 public:
00119   virtual ~DocCheckboxObject() override = default;
00120
00122   static const char* ObjectTypeStatic();
00123
00125   virtual const se::common::OcrString& GetOcrString() const = 0;
00127   virtual const se::common::OcrString* GetOcrStringPtr() const = 0;
00128
00129 public:
00133
```

```
00135   virtual se::common::OcrString& GetMutableOcrString() = 0;
00137   virtual se::common::OcrString* GetMutableOcrStringPtr() = 0;
00139   virtual void SetOcrString(const se::common::OcrString& ocrstring) = 0;
00140 };
00141
00142
00146 class SE_DLL_EXPORT DocTemplateObject : public DocBasicObject {
00147 public:
00149   virtual ~DocTemplateObject() override = default;
00150
00152   static const char* ObjectTypeStatic();
00153 };
00154
00155
00159 class SE_DLL_EXPORT DocLineObject : public DocBasicObject {
00160 public:
00162   virtual ~DocLineObject() override = default;
00163
00165   static const char* ObjectTypeStatic();
00166 };
00167
00168
00173 class SE_DLL_EXPORT DocZoneObject : public DocBasicObject {
00174 public:
00176   virtual ~DocZoneObject() override = default;
00177
00179   static const char* ObjectTypeStatic();
00180
00182   virtual const se::common::Size& GetSize() const = 0;
00184   virtual se::common::Size& GetMutableSize() = 0;
00186   virtual const se::common::Size* GetSizePtr() const = 0;
00188   virtual se::common::Size* GetMutableSizePtr() = 0;
00190   virtual void SetSize(const se::common::Size& size) = 0;
00191 };
00192
00193
00198 class SE_DLL_EXPORT DocMultiStringTextObject : public DocBasicObject {
00199 public:
00201   virtual ~DocMultiStringTextObject() override = default;
00202
00204   static const char* ObjectTypeStatic();
00205
00207   virtual int GetStringsCount() const = 0;
00209   virtual void SetStringsCount(int count) = 0;
00210
00212   virtual const DocTextObject& GetStringObject(int index) const = 0;
00214   virtual DocTextObject& GetMutableStringObject(int index) = 0;
00216   virtual const DocTextObject* GetStringObjectPtr(int index) const = 0;
00218   virtual DocTextObject* GetMutableStringObjectPtr(int index) = 0;
00220   virtual void SetStringObject(
00221       int index, const DocTextObject& text_object) = 0;
00222 };
00223
00224
00228 class SE_DLL_EXPORT DocMetaObject : public DocBasicObject {
00229 public:
00231   virtual ~DocMetaObject() override = default;
00232
00234   static const char* ObjectTypeStatic();
00235
00237   virtual const se::common::OcrString& GetOcrString() const = 0;
00239   virtual const se::common::OcrString* GetOcrStringPtr() const = 0;
00240
00241 public:
00245
00247   virtual se::common::OcrString& GetMutableOcrString() = 0;
00249   virtual se::common::OcrString* GetMutableOcrStringPtr() = 0;
00251   virtual void SetOcrString(const se::common::OcrString& ocrstring) = 0;
00252 };
00253
00254
00258 class SE_DLL_EXPORT DocTableObject : public DocBasicObject {
00259 public:
00261   virtual ~DocTableObject() override = default;
00262
00264   static const char* ObjectTypeStatic();
00265
00267   virtual int GetRowsCount() const = 0;
00270   virtual int GetColsCount(int row) const = 0;
00272   virtual void ResizeRows(int rows) = 0;
00274   virtual void ResizeCols(int row, int cols) = 0;
00275
00277   virtual const char* GetColName(int col, int row) const = 0;
00280   virtual void SetColName(int col, int first_row, const char* col_name) = 0;
00281
00283   virtual const DocTextObject& GetTextCell(int row, int col) const = 0;
00285   virtual const DocTextObject* GetTextCellPtr(int row, int col) const = 0;
```

```
00286
00287 public:
00291
00293   virtual const DocMultiStringTextObject& GetCell(int row, int col) const = 0;
00295   virtual DocMultiStringTextObject& GetMutableCell(int row, int col) = 0;
00297   virtual const DocMultiStringTextObject* GetCellPtr(int row, int col) const = 0;
00299   virtual DocMultiStringTextObject* GetMutableCellPtr(int row, int col) = 0;
00301   virtual void SetCell(
00302     int row,
00303     int col,
00304     const DocMultiStringTextObject& multi_string_text_object) = 0;
00305 };
00306
00307
00311 class SE_DLL_EXPORT DocImageObject : public DocBasicObject {
00312 public:
00314   virtual ~DocImageObject() override = default;
00315
00317   static const char* ObjectTypeStatic();
00318 };
00319
00320
00324 class SE_DLL_EXPORT DocBarcodeObject : public DocBasicObject {
00325 public:
00327   virtual ~DocBarcodeObject() override = default;
00328
00330   static const char* ObjectTypeStatic();
00331
00333   virtual const se::common::MutableString& GetDecodedString() const = 0;
00335   virtual const se::common::MutableString* GetDecodedStringPtr() const = 0;
00336
00337
00338 public:
00342
00344   virtual se::common::MutableString& GetMutableDecodedString() = 0;
00346   virtual se::common::MutableString* GetMutableDecodedStringPtr() = 0;
00348   virtual void SetDecodedString(const se::common::MutableString& decstring) = 0;
00349 };
00350
00354 class SE_DLL_EXPORT DocMarkObject : public DocBasicObject {
00355 public:
00357   virtual ~DocMarkObject() override = default;
00358
00360   static const char* ObjectTypeStatic();
00361 };
00362
00363
00364 } } // namespace se::doc
00365
00366 #endif // DOCENGINE_DOC_OBJECTS_H_INCLUDED
```

2.29 doc_objects_collection.h File Reference

Collection of basic objects for Smart Document Engine.

Classes

- class se::doc::DocObjectsCollection

  The class representing a collection of graphical objects CLASS TO BE DEPRECATED.

2.29.1 Detailed Description

Collection of basic objects for Smart Document Engine.

Definition in file doc_objects_collection.h.

2.30 doc_objects_collection.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_OBJECTS_COLLECTION_H_INCLUDED
00012 #define DOCENGINE_DOC_OBJECTS_COLLECTION_H_INCLUDED
00013
00014 #include <secommon/se_serialization.h>
00015 #include <secommon/se_export_defs.h>
00016
00017 #include <docengine/doc_forward_declarations.h>
00018 #include <docengine/doc_basic_objects_iterator.h>
00019
00020
00021 namespace se { namespace doc {
00022
00023
00028 class SE_DLL_EXPORT DocObjectsCollection {
00029 public:
00031   static const char* BaseClassNameStatic();
00032
00033 public:
00041   static DocObjectsCollection* Create(const char* object_type);
00042
00049   virtual DocBasicObject* CreateObject() const = 0;
00050
00051 public:
00053   virtual ~DocObjectsCollection() = default;
00054
00060   virtual DocObjectsCollection* Clone() const = 0;
00061
00063   virtual const char* ObjectType() const = 0;
00064
00066   virtual int GetFrameID() const = 0;
00068   virtual void SetFrameID(int frame_id) = 0;
00069
00071   virtual int GetObjectsCount() const = 0;
00073   virtual bool HasObject(int obj_id) const = 0;
00075   virtual const DocBasicObject& GetObject(int obj_id) const = 0;
00077   virtual DocBasicObject& GetMutableObject(int obj_id) = 0;
00079   virtual const DocBasicObject* GetObjectPtr(int obj_id) const = 0;
00081   virtual DocBasicObject* GetMutableObjectPtr(int obj_id) = 0;
00083   virtual const DocTagsCollection& GetObjectTags(int obj_id) const = 0;
00085   virtual const DocTagsCollection* GetObjectTagsPtr(int obj_id) const = 0;
00087   virtual DocBasicObjectsMutableIterator AddObject(
00088       const DocBasicObject& obj) = 0;
00090   virtual void SetObject(int obj_id, const DocBasicObject& obj) = 0;
00092   virtual void RemoveObject(int obj_id) = 0;
00095   virtual void RemoveObjectDeep(
00096       int              obj_id,
00097       DocViewsCollection& views_collection) = 0;
00098
00100   virtual DocBasicObjectsIterator BasicObjectsBegin() const = 0;
00102   virtual DocBasicObjectsIterator BasicObjectsEnd() const = 0;
00103
00105   virtual DocBasicObjectsMutableIterator MutableBasicObjectsBegin() = 0;
00107   virtual DocBasicObjectsMutableIterator MutableBasicObjectsEnd() = 0;
00108
00110   virtual DocBasicObjectsSliceIterator BasicObjectsSlice(
00111       const char* tag) const = 0;
00112
00114   virtual DocBasicObjectsMutableSliceIterator MutableBasicObjectsSlice(
00115       const char* tag) = 0;
00116
00118   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00119 };
00120
00121
00122 } } // namespace se::doc
00123
00124 #endif // DOCENGINE_DOC_OBJECTS_COLLECTION_H_INCLUDED
```

2.31 doc_objects_collections_iterator.h File Reference

Smart Document Engine basic graphical objects collections iterator.

Classes

- class se::doc::DocObjectsCollectionsIterator

  Basic const-ref iterator for graphical object collections CLASS TO BE DEPRECATED.
- class se::doc::DocObjectsCollectionsMutableIterator

  Mutable-ref iterator for graphical object collections.
- class se::doc::DocObjectsCollectionsSliceIterator

  Const-ref iterator for graphical object collections with a given tag.
- class se::doc::DocObjectsCollectionsMutableSliceIterator

  Const-ref iterator for object collections with a given tag.

2.31.1 Detailed Description

Smart Document Engine basic graphical objects collections iterator.

Definition in file doc_objects_collections_iterator.h.

2.32 doc_objects_collections_iterator.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_OBJECTS_COLLECTIONS_ITERATOR_H_INCLUDED
00012 #define DOCENGINE_DOC_OBJECTS_COLLECTIONS_ITERATOR_H_INCLUDED
00013
00014 #include <secommon/se_export_defs.h>
00015 #include <docengine/doc_forward_declarations.h>
00016
00017 namespace se { namespace doc {
00018
00021 class DocObjectsCollectionsIteratorImpl;
00022
00027 class SE_DLL_EXPORT DocObjectsCollectionsIterator {
00028 private:
00030   DocObjectsCollectionsIterator(
00031       const DocObjectsCollectionsIteratorImpl& pimpl);
00032
00033 public:
00035   DocObjectsCollectionsIterator(const DocObjectsCollectionsIterator& other);
00037   DocObjectsCollectionsIterator& operator =(
00038       const DocObjectsCollectionsIterator& other);
00040   ~DocObjectsCollectionsIterator();
00041
00043   static DocObjectsCollectionsIterator ConstructFromImpl(
00044       const DocObjectsCollectionsIteratorImpl& pimpl);
00045
00047   int GetID() const;
00049   const DocObjectsCollection& GetObjectsCollection() const;
00051   const DocTagsCollection& GetTags() const;
00053   const DocObjectsCollection* GetObjectsCollectionPtr() const;
00055   const DocTagsCollection* GetTagsPtr() const;
00057   void Advance();
00058
00060   bool Equals(const DocObjectsCollectionsIterator& rvalue) const;
00062   bool operator ==(const DocObjectsCollectionsIterator& rvalue) const;
00064   bool operator !=(const DocObjectsCollectionsIterator& rvalue) const;
00065
00066 private:
00068   DocObjectsCollectionsIteratorImpl* pimpl_;
00069 };
00070
00071
00074 class DocObjectsCollectionsMutableIteratorImpl;
00075
00079 class SE_DLL_EXPORT DocObjectsCollectionsMutableIterator {
00080 private:
00082   DocObjectsCollectionsMutableIterator(
```

```
00083       const DocObjectsCollectionsMutableIteratorImpl& pimpl);
00084
00085 public:
00087   DocObjectsCollectionsMutableIterator(
00088       const DocObjectsCollectionsMutableIterator& other);
00090   DocObjectsCollectionsMutableIterator& operator =(
00091       const DocObjectsCollectionsMutableIterator& other);
00093   ~DocObjectsCollectionsMutableIterator();
00094
00096   static DocObjectsCollectionsMutableIterator ConstructFromImpl(
00097       const DocObjectsCollectionsMutableIteratorImpl& pimpl);
00098
00100   int GetID() const;
00102   const DocObjectsCollection& GetObjectsCollection() const;
00104   DocObjectsCollection& GetMutableObjectsCollection() const;
00106   const DocTagsCollection& GetTags() const;
00107
00109   const DocObjectsCollection* GetObjectsCollectionPtr() const;
00111   DocObjectsCollection* GetMutableObjectsCollectionPtr() const;
00113   const DocTagsCollection* GetTagsPtr() const;
00115   void Advance();
00116
00118   bool Equals(const DocObjectsCollectionsMutableIterator& rvalue) const;
00120   bool operator ==(const DocObjectsCollectionsMutableIterator& rvalue) const;
00122   bool operator !=(const DocObjectsCollectionsMutableIterator& rvalue) const;
00123
00124 private:
00126   DocObjectsCollectionsMutableIteratorImpl* pimpl_;
00127 };
00128
00129
00132 class DocObjectsCollectionsSliceIteratorImpl;
00133
00134
00138 class SE_DLL_EXPORT DocObjectsCollectionsSliceIterator {
00139 private:
00141   DocObjectsCollectionsSliceIterator(
00142       const DocObjectsCollectionsSliceIteratorImpl& pimpl);
00143
00144 public:
00146   DocObjectsCollectionsSliceIterator(
00147       const DocObjectsCollectionsSliceIterator& other);
00149   DocObjectsCollectionsSliceIterator& operator =(
00150       const DocObjectsCollectionsSliceIterator& other);
00152   ~DocObjectsCollectionsSliceIterator();
00153
00155   static DocObjectsCollectionsSliceIterator ConstructFromImpl(
00156       const DocObjectsCollectionsSliceIteratorImpl& pimpl);
00157
00159   int GetID() const;
00161   const DocObjectsCollection& GetObjectsCollection() const;
00163   const DocTagsCollection& GetTags() const;
00165   const DocObjectsCollection* GetObjectsCollectionPtr() const;
00167   const DocTagsCollection* GetTagsPtr() const;
00169   void Advance();
00170
00173   bool Finished() const;
00174
00175 private:
00177   DocObjectsCollectionsSliceIteratorImpl* pimpl_;
00178 };
00179
00180
00183 class DocObjectsCollectionsMutableSliceIteratorImpl;
00184
00185
00189 class SE_DLL_EXPORT DocObjectsCollectionsMutableSliceIterator {
00190 private:
00192   DocObjectsCollectionsMutableSliceIterator(
00193       const DocObjectsCollectionsMutableSliceIteratorImpl& pimpl);
00194
00195 public:
00197   DocObjectsCollectionsMutableSliceIterator(
00198       const DocObjectsCollectionsMutableSliceIterator& other);
00200   DocObjectsCollectionsMutableSliceIterator& operator =(
00201       const DocObjectsCollectionsMutableSliceIterator& other);
00203   ~DocObjectsCollectionsMutableSliceIterator();
00204
00206   static DocObjectsCollectionsMutableSliceIterator ConstructFromImpl(
00207       const DocObjectsCollectionsMutableSliceIteratorImpl& pimpl);
00208
00210   int GetID() const;
00212   const DocObjectsCollection& GetObjectsCollection() const;
00214   DocObjectsCollection& GetMutableObjectsCollection() const;
00216   const DocTagsCollection& GetTags() const;
00218   const DocObjectsCollection* GetObjectsCollectionPtr() const;
00220   DocObjectsCollection* GetMutableObjectsCollectionPtr() const;
```

```
00222   const DocTagsCollection* GetTagsPtr() const;
00224   void Advance();
00225
00228   bool Finished() const;
00229
00230 private:
00232   DocObjectsCollectionsMutableSliceIteratorImpl* pimpl_;
00233 };
00234
00235
00236 } } // namespace se::doc
00237
00238 #endif // DOCENGINE_DOC_OBJECTS_COLLECTIONS_ITERATOR_H_INCLUDED
```

2.33 doc_physical_document.h File Reference

Smart Document Engine class for document pages processing result and graphical objects extraction.

Classes

- class se::doc::DocPageInfo

  The additional infomation about a processed physical page.
- class se::doc::DocPhysicalPage

  The class representing the found physical page.
- class se::doc::DocPhysicalDocument

  The class representing the found physical document.

2.33.1 Detailed Description

Smart Document Engine class for document pages processing result and graphical objects extraction.

Definition in file doc_physical_document.h.

2.34 doc_physical_document.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_PHYSICAL_DOCUMENT_H_INCLUDED
00012 #define DOCENGINE_DOC_PHYSICAL_DOCUMENT_H_INCLUDED
00013
00014 #include <secommon/se_common.h>
00015 #include <docengine/doc_basic_objects_iterator.h>
00016
00017 #ifdef WITH_PDFCREATOR
00018 #include <secommon/se_pdf_creator.h>
00019 #endif
00020
00021 namespace se { namespace doc {
00022
00023
00027 class SE_DLL_EXPORT DocPageInfo {
00028 public:
00030   virtual ~DocPageInfo() = default;
00031
00033   virtual bool IsGarbage() const = 0;
00034
00036   virtual int GarbageReasonsCount() const = 0;
00037
00039   virtual const char* GarbageReason(int idx) const = 0;
00040 };
```

```
00041
00042
00046 class SE_DLL_EXPORT DocPhysicalPage {
00047 public:
00049   virtual ~DocPhysicalPage() = default;
00050
00052   virtual int GetSourceSceneID() const = 0;
00053
00055   virtual DocTextObjectsIterator TextObjectsBegin(const char* name) const = 0;
00057   virtual DocTextObjectsIterator TextObjectsEnd(const char* name) const = 0;
00059   virtual DocImageObjectsIterator ImageObjectsBegin(const char* name) const = 0;
00061   virtual DocImageObjectsIterator ImageObjectsEnd(const char* name) const = 0;
00063   virtual DocTableObjectsIterator TableObjectsBegin(const char* name) const = 0;
00065   virtual DocTableObjectsIterator TableObjectsEnd(const char* name) const = 0;
00067   virtual DocBarcodeObjectsIterator BarcodeObjectsBegin(const char* name) const = 0;
00069   virtual DocBarcodeObjectsIterator BarcodeObjectsEnd(const char* name) const = 0;
00071   virtual DocCheckboxObjectsIterator CheckboxObjectsBegin(const char* name) const = 0;
00073   virtual DocCheckboxObjectsIterator CheckboxObjectsEnd(const char* name) const = 0;
00075   virtual DocMetaObjectsIterator ForensicObjectsBegin(const char* name) const = 0;
00077   virtual DocMetaObjectsIterator ForensicObjectsEnd(const char* name) const = 0;
00079   virtual DocForensicCheckObjectsIterator ForensicCheckObjectsBegin(const char* name) const = 0;
00081   virtual DocForensicCheckObjectsIterator ForensicCheckObjectsEnd(const char* name) const = 0;
00083   virtual int GetTextObjectsCount(const char* name) const = 0;
00085   virtual int GetImageObjectsCount(const char* name) const = 0;
00087   virtual int GetTableObjectsCount(const char* name) const = 0;
00089   virtual int GetBarcodeObjectsCount(const char* name) const = 0;
00091   virtual int GetCheckboxObjectsCount(const char* name) const = 0;
00093   virtual int GetForensicObjectsCount(const char* name) const = 0;
00095   virtual int GetForensicCheckObjectsCount(const char* name) const = 0;
00096
00098   virtual bool HasBasicObjects() const = 0;
00099
00101   virtual const DocPageInfo& GetPageInfo() const = 0;
00103   virtual const DocPageInfo* GetPageInfoPtr() const = 0;
00104
00106   virtual const se::common::Quadrangle& GetPageQuadrangle() const = 0;
00108   virtual const se::common::Polygon& GetPagePolygon() const = 0;
00110   virtual const se::common::Quadrangle* GetPageQuadranglePtr() const = 0;
00112   virtual const se::common::Polygon* GetPagePolygonPtr() const = 0;
00113
00115   virtual DocTextObjectsIterator GetFulltextBasicObjectsBegin() const = 0;
00117   virtual DocTextObjectsIterator GetFulltextBasicObjectsEnd() const = 0;
00118
00120   virtual se::common::Image* GetPageImageFromScene(const se::common::Image& scene_image) const = 0;
00121
00123   virtual DocTextObjectsIterator RawTextObjectsBegin() const = 0;
00125   virtual DocTextObjectsIterator RawTextObjectsEnd() const = 0;
00127   virtual int GetRawTextObjectsCount() const = 0;
00129   virtual bool HasRawTextObject(const char* name) const = 0;
00131   virtual const se::doc::DocTextObject& GetRawTextObject(const char* name) const = 0;
00132
00133 #ifdef WITH_PDFCREATOR
00135   virtual void FillPDFContainer(se::pdf::PdfSourcesContainer* pdf_container, const se::common::Image&
      scene_image) const = 0;
00136 #endif
00137
00138 public:
00142
00144   virtual DocBasicObjectsIterator BasicObjectsBegin(const char* name) const = 0;
00146   virtual DocBasicObjectsIterator BasicObjectsEnd(const char* name) const = 0;
00148   virtual int GetBasicObjectsCount(const char* name) const = 0;
00149 };
00150
00151
00155 class SE_DLL_EXPORT DocPhysicalDocument {
00156 public:
00158   virtual ~DocPhysicalDocument() = default;
00159
00161   virtual int GetTextObjectsCount(const char* name) const = 0;
00163   virtual int GetTableObjectsCount(const char* name) const = 0;
00165   virtual int GetImageObjectsCount(const char* name) const = 0;
00167   virtual int GetForensicObjectsCount(const char* name) const = 0;
00169   virtual int GetForensicCheckObjectsCount(const char* name) const = 0;
00171   virtual int GetBarcodeObjectsCount(const char* name) const = 0;
00173   virtual int GetCheckboxObjectsCount(const char* name) const = 0;
00174
00176   virtual int GetPagesCount() const = 0;
00177
00179   virtual const DocPhysicalPage& GetPhysicalPage(int idx) const = 0;
00181   virtual const DocPhysicalPage* GetPhysicalPagePtr(int idx) const = 0;
00182
00183 public:
00187
00189   virtual int GetBasicObjectsCount(const char* name) const = 0;
00190 };
00191
00192 }} //se::doc
```

```
00193
00194 #endif //DOCENGINE_DOC_PHYSICAL_DOCUMENT_H_INCLUDED
```

2.35 doc_physical_document_iterators.h File Reference

Smart Document Engine class for iterators used in document pages processing result and graphical objects extraction.

Classes

- class se::doc::DocTextObjectsCrossPageIterator

  Basic const-ref iterator for a collection of text objects from several pages.
- class se::doc::DocForensicCheckObjectsCrossPageIterator

  Basic const-ref iterator for a collection of forensic check objects from several pages.
- class se::doc::DocImageObjectsCrossPageIterator

  Basic const-ref iterator for a collection of image objects from several pages.
- class se::doc::DocTableObjectsCrossPageIterator

  Basic const-ref iterator for a collection of table objects from several pages.
- class se::doc::DocCheckboxObjectsCrossPageIterator

  Basic const-ref iterator for a collection of checkbox objects from several pages.
- class se::doc::DocMetaObjectsCrossPageIterator

  Basic const-ref iterator for a collection of meta objects from several pages.
- class se::doc::DocBarcodeObjectsCrossPageIterator

  Basic const-ref iterator for a collection of barcode objects from several pages.

2.35.1 Detailed Description

Smart Document Engine class for iterators used in document pages processing result and graphical objects extraction.

Definition in file doc_physical_document_iterators.h.

2.36 doc_physical_document_iterators.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_PHYSICAL_DOCUMENT_ITERATORS_H_INCLUDED
00012 #define DOCENGINE_DOC_PHYSICAL_DOCUMENT_ITERATORS_H_INCLUDED
00013
00014 #include <secommon/se_common.h>
00015 #include <docengine/doc_basic_object.h>
00016
00017 namespace se { namespace doc {
00018
00021 //class DocBasicObjectsCrossPageIteratorImpl;
00022
00023 class DocTextObjectsCrossPageIteratorImpl;
00024 class DocForensicCheckObjectsCrossPageIteratorImpl;
00025 class DocImageObjectsCrossPageIteratorImpl;
00026 class DocTableObjectsCrossPageIteratorImpl;
00027 class DocCheckboxObjectsCrossPageIteratorImpl;
00028 class DocMetaObjectsCrossPageIteratorImpl;
00029 class DocBarcodeObjectsCrossPageIteratorImpl;
```

```
00030
00031
00036 class SE_DLL_EXPORT DocTextObjectsCrossPageIterator {
00037 private:
00039     DocTextObjectsCrossPageIterator(const DocTextObjectsCrossPageIteratorImpl& pimpl);
00040
00041 public:
00043     DocTextObjectsCrossPageIterator(const DocTextObjectsCrossPageIterator& other);
00045     DocTextObjectsCrossPageIterator& operator =(const DocTextObjectsCrossPageIterator& other);
00047     ~DocTextObjectsCrossPageIterator();
00048
00050     static DocTextObjectsCrossPageIterator ConstructFromImpl(
00051         const DocTextObjectsCrossPageIteratorImpl& pimpl);
00052
00054     int GetPhysicalPageID() const;
00055
00057     int GetObjectID() const;
00058
00060     const DocTextObject& GetTextObject() const;
00062     const DocTextObject* GetTextObjectPtr() const;
00064     void Advance();
00065
00067     bool Equals(const DocTextObjectsCrossPageIterator& rvalue) const;
00069     bool operator ==(const DocTextObjectsCrossPageIterator& rvalue) const;
00071     bool operator !=(const DocTextObjectsCrossPageIterator& rvalue) const;
00072
00073 private:
00075     DocTextObjectsCrossPageIteratorImpl* pimpl_;
00076 };
00077
00082 class SE_DLL_EXPORT DocForensicCheckObjectsCrossPageIterator {
00083 private:
00085     DocForensicCheckObjectsCrossPageIterator(const DocForensicCheckObjectsCrossPageIteratorImpl&
    pimpl);
00086
00087 public:
00089     DocForensicCheckObjectsCrossPageIterator(const DocForensicCheckObjectsCrossPageIterator& other);
00091     DocForensicCheckObjectsCrossPageIterator& operator =(const
    DocForensicCheckObjectsCrossPageIterator& other);
00093     ~DocForensicCheckObjectsCrossPageIterator();
00094
00096     static DocForensicCheckObjectsCrossPageIterator ConstructFromImpl(
00097         const DocForensicCheckObjectsCrossPageIteratorImpl& pimpl);
00098
00100     int GetPhysicalPageID() const;
00101
00103     const DocForensicCheckObject& GetForensicCheckObject() const;
00105     const DocForensicCheckObject* GetForensicCheckObjectPtr() const;
00107     void Advance();
00108
00110     bool Equals(const DocForensicCheckObjectsCrossPageIterator& rvalue) const;
00112     bool operator ==(const DocForensicCheckObjectsCrossPageIterator& rvalue) const;
00114     bool operator !=(const DocForensicCheckObjectsCrossPageIterator& rvalue) const;
00115
00116 private:
00118     DocForensicCheckObjectsCrossPageIteratorImpl* pimpl_;
00119 };
00120
00125 class SE_DLL_EXPORT DocImageObjectsCrossPageIterator {
00126 private:
00128     DocImageObjectsCrossPageIterator(const DocImageObjectsCrossPageIteratorImpl& pimpl);
00129
00130 public:
00132     DocImageObjectsCrossPageIterator(const DocImageObjectsCrossPageIterator& other);
00134     DocImageObjectsCrossPageIterator& operator =(const DocImageObjectsCrossPageIterator& other);
00136     ~DocImageObjectsCrossPageIterator();
00137
00139     static DocImageObjectsCrossPageIterator ConstructFromImpl(
00140         const DocImageObjectsCrossPageIteratorImpl& pimpl);
00141
00143     int GetPhysicalPageID() const;
00144
00146     int GetObjectID() const;
00147
00149     const DocImageObject& GetImageObject() const;
00151     const DocImageObject* GetImageObjectPtr() const;
00153     void Advance();
00154
00156     bool Equals(const DocImageObjectsCrossPageIterator& rvalue) const;
00158     bool operator ==(const DocImageObjectsCrossPageIterator& rvalue) const;
00160     bool operator !=(const DocImageObjectsCrossPageIterator& rvalue) const;
00161
00162 private:
00164     DocImageObjectsCrossPageIteratorImpl* pimpl_;
00165 };
00166
00171 class SE_DLL_EXPORT DocTableObjectsCrossPageIterator {
```

```
00172 private:
00174     DocTableObjectsCrossPageIterator(const DocTableObjectsCrossPageIteratorImpl& pimpl);
00175
00176 public:
00178     DocTableObjectsCrossPageIterator(const DocTableObjectsCrossPageIterator& other);
00180     DocTableObjectsCrossPageIterator& operator =(const DocTableObjectsCrossPageIterator& other);
00182     ~DocTableObjectsCrossPageIterator();
00183
00185     static DocTableObjectsCrossPageIterator ConstructFromImpl(
00186         const DocTableObjectsCrossPageIteratorImpl& pimpl);
00187
00189     int GetPhysicalPageID() const;
00190
00192     int GetObjectID() const;
00193
00195     const DocTableObject& GetTableObject() const;
00197     const DocTableObject* GetTableObjectPtr() const;
00199     void Advance();
00200
00202     bool Equals(const DocTableObjectsCrossPageIterator& rvalue) const;
00204     bool operator ==(const DocTableObjectsCrossPageIterator& rvalue) const;
00206     bool operator !=(const DocTableObjectsCrossPageIterator& rvalue) const;
00207
00208 private:
00210     DocTableObjectsCrossPageIteratorImpl* pimpl_;
00211 };
00212
00217 class SE_DLL_EXPORT DocCheckboxObjectsCrossPageIterator {
00218 private:
00220     DocCheckboxObjectsCrossPageIterator(const DocCheckboxObjectsCrossPageIteratorImpl& pimpl);
00221
00222 public:
00224     DocCheckboxObjectsCrossPageIterator(const DocCheckboxObjectsCrossPageIterator& other);
00226     DocCheckboxObjectsCrossPageIterator& operator =(const DocCheckboxObjectsCrossPageIterator& other);
00228     ~DocCheckboxObjectsCrossPageIterator();
00229
00231     static DocCheckboxObjectsCrossPageIterator ConstructFromImpl(
00232         const DocCheckboxObjectsCrossPageIteratorImpl& pimpl);
00233
00235     int GetPhysicalPageID() const;
00236
00238     int GetObjectID() const;
00239
00241     const DocCheckboxObject& GetCheckboxObject() const;
00243     const DocCheckboxObject* GetCheckboxObjectPtr() const;
00245     void Advance();
00246
00248     bool Equals(const DocCheckboxObjectsCrossPageIterator& rvalue) const;
00250     bool operator ==(const DocCheckboxObjectsCrossPageIterator& rvalue) const;
00252     bool operator !=(const DocCheckboxObjectsCrossPageIterator& rvalue) const;
00253
00254 private:
00256     DocCheckboxObjectsCrossPageIteratorImpl* pimpl_;
00257 };
00258
00263 class SE_DLL_EXPORT DocMetaObjectsCrossPageIterator {
00264 private:
00266     DocMetaObjectsCrossPageIterator(const DocMetaObjectsCrossPageIteratorImpl& pimpl);
00267
00268 public:
00270     DocMetaObjectsCrossPageIterator(const DocMetaObjectsCrossPageIterator& other);
00272     DocMetaObjectsCrossPageIterator& operator =(const DocMetaObjectsCrossPageIterator& other);
00274     ~DocMetaObjectsCrossPageIterator();
00275
00277     static DocMetaObjectsCrossPageIterator ConstructFromImpl(
00278         const DocMetaObjectsCrossPageIteratorImpl& pimpl);
00279
00281     int GetPhysicalPageID() const;
00282
00284     const DocMetaObject& GetMetaObject() const;
00286     const DocMetaObject* GetMetaObjectPtr() const;
00288     void Advance();
00289
00291     bool Equals(const DocMetaObjectsCrossPageIterator& rvalue) const;
00293     bool operator ==(const DocMetaObjectsCrossPageIterator& rvalue) const;
00295     bool operator !=(const DocMetaObjectsCrossPageIterator& rvalue) const;
00296
00297 private:
00299     DocMetaObjectsCrossPageIteratorImpl* pimpl_;
00300 };
00301
00306 class SE_DLL_EXPORT DocBarcodeObjectsCrossPageIterator {
00307 private:
00309     DocBarcodeObjectsCrossPageIterator(const DocBarcodeObjectsCrossPageIteratorImpl& pimpl);
00310
00311 public:
00313     DocBarcodeObjectsCrossPageIterator(const DocBarcodeObjectsCrossPageIterator& other);
```

```
00315     DocBarcodeObjectsCrossPageIterator& operator =(const DocBarcodeObjectsCrossPageIterator& other);
00317     ~DocBarcodeObjectsCrossPageIterator();
00318
00320     static DocBarcodeObjectsCrossPageIterator ConstructFromImpl(
00321         const DocBarcodeObjectsCrossPageIteratorImpl& pimpl);
00322
00324     int GetPhysicalPageID() const;
00325
00327     int GetObjectID() const;
00328
00330     const DocBarcodeObject& GetBarcodeObject() const;
00332     const DocBarcodeObject* GetBarcodeObjectPtr() const;
00334     void Advance();
00335
00337     bool Equals(const DocBarcodeObjectsCrossPageIterator& rvalue) const;
00339     bool operator ==(const DocBarcodeObjectsCrossPageIterator& rvalue) const;
00341     bool operator !=(const DocBarcodeObjectsCrossPageIterator& rvalue) const;
00342
00343 private:
00345     DocBarcodeObjectsCrossPageIteratorImpl* pimpl_;
00346 };
00347
00348
00349 }} //se::doc
00350
00351 #endif //DOCENGINE_DOC_PHYSICAL_DOCUMENT_ITERATORS_H_INCLUDED
```

2.37 doc_processing_settings.h File Reference

Smart Document Engine source processing settings.

Classes

- class se::doc::DocProcessingSettings

  The class representing the settings of a single processing iteration.

2.37.1 Detailed Description

Smart Document Engine source processing settings.

Definition in file doc_processing_settings.h.

2.38 doc_processing_settings.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_PROCESSING_SETTINGS_H_INCLUDED
00012 #define DOCENGINE_DOC_PROCESSING_SETTINGS_H_INCLUDED
00013
00014 #include <secommon/se_common.h>
00015 #include <docengine/doc_feedback.h>
00016 #include <docengine/doc_forward_declarations.h>
00017
00018 namespace se { namespace doc {
00019
00023 class SE_DLL_EXPORT DocProcessingSettings {
00024 public:
00026   virtual ~DocProcessingSettings() = default;
00027
00029   virtual int GetOptionsCount() const = 0;
00031   virtual bool HasOption(const char* option_name) const = 0;
00033   virtual const char* GetOption(const char* option_name) const = 0;
00035   virtual void SetOption(const char* option_name, const char* option_value) = 0;
```

```
00037   virtual void RemoveOption(const char* option_name) = 0;
00039   virtual se::common::StringsMapIterator OptionsBegin() const = 0;
00041   virtual se::common::StringsMapIterator OptionsEnd() const = 0;
00042
00044   virtual int GetSessionOptionsCount() const = 0;
00046   virtual bool HasSessionOption(const char* option_name) const = 0;
00048   virtual const char* GetSessionOption(const char* option_name) const = 0;
00050   virtual se::common::StringsMapIterator SessionOptionsBegin() const = 0;
00052   virtual se::common::StringsMapIterator SessionOptionsEnd() const = 0;
00053
00055   virtual int GetEnabledDocumentTypesCount() const = 0;
00057   virtual bool HasEnabledDocumentType(const char* doc_name) const = 0;
00059   virtual const char* GetEnabledDocumentType(int doc_id) const = 0;
00060
00061 public:
00065
00067   virtual int GetCurrentSourceID() const = 0;
00069   virtual void SetCurrentSourceID(int source_id) = 0;
00071   virtual int GetAvailableRoutinesCount() const = 0;
00073   virtual bool HasAvailableRoutine(const char* routine_name) const = 0;
00075   virtual se::common::StringsMapIterator AvailableRoutinesBegin() const = 0;
00077   virtual se::common::StringsMapIterator AvailableRoutinesEnd() const = 0;
00078
00080   virtual int RoutinesQueueSize() const = 0;
00082   virtual const char* RoutinesQueueFront() const = 0;
00084   virtual void RoutinesQueuePush(const char* routine_name) = 0;
00086   virtual void RoutinesQueuePop() = 0;
00088   virtual void RoutinesQueueClear() = 0;
00089
00091   virtual void BindFeedbackReporter(DocFeedback* feedback_reporter) = 0;
00093   virtual DocFeedback* GetFeedbackReporter() const = 0;
00094
00095 };
00096
00097
00098 } } // namespace se::doc
00099
00100 #endif // DOCENGINE_DOC_PROCESSING_SETTINGS_H_INCLUDED
```

2.39 doc_result.h File Reference

Smart Document Engine result representation.

Classes

- class se::doc::DocResult

  The class representing the document analysis and recognition result.

2.39.1 Detailed Description

Smart Document Engine result representation.

Definition in file doc_result.h.

2.40 doc_result.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_RESULT_H_INCLUDED
00012 #define DOCENGINE_DOC_RESULT_H_INCLUDED
00013
00014 #include <secommon/se_common.h>
00015 #include <docengine/doc_forward_declarations.h>
```

```
00016 #include <docengine/doc_documents_iterator.h>
00017 #include <docengine/doc_physical_document.h>
00018 #include <docengine/doc_scene_info.h>
00019
00020 namespace se { namespace doc {
00021
00025 class SE_DLL_EXPORT DocResult {
00026 public:
00028   virtual ~DocResult() = default;
00029
00031   virtual DocResult* PartialClone() const = 0;
00033   virtual DocResult* Clone() const = 0;
00034
00035
00037   virtual int GetDocumentsCount() const = 0;
00039   virtual bool HasDocument(int doc_id) const = 0;
00041   virtual const Document& GetDocument(int doc_id) const = 0;
00043   virtual const Document* GetDocumentPtr(int doc_id) const = 0;
00044
00046   virtual DocumentsIterator DocumentsBegin() const = 0;
00048   virtual DocumentsIterator DocumentsEnd() const = 0;
00049
00051   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00052
00054   virtual const DocPhysicalDocument& GetPhysicalDocument(int idx) const = 0;
00055
00057   virtual const DocPhysicalDocument* GetPhysicalDocumentPtr(int idx) const = 0;
00058
00060   virtual int GetScenesCount() const = 0;
00061
00063   virtual const DocSceneInfo& GetSceneInfo(int idx) const = 0;
00064
00066   virtual const DocSceneInfo& GetLastSceneInfo() const = 0;
00067
00069   virtual const DocSceneInfo* GetSceneInfoPtr(int idx) const = 0;
00070
00072   virtual const DocSceneInfo* GetLastSceneInfoPtr() const = 0;
00073
00074 public:
00078
00080   virtual const DocGraphicalStructure& GetGraphicalStructure() const = 0;
00082   virtual DocGraphicalStructure& GetMutableGraphicalStructure() = 0;
00084   virtual const DocGraphicalStructure* GetGraphicalStructurePtr() const = 0;
00086   virtual DocGraphicalStructure* GetMutableGraphicalStructurePtr() = 0;
00087
00089   virtual Document& GetMutableDocument(int doc_id) = 0;
00091   virtual const DocTagsCollection& GetDocumentTags(int doc_id) const = 0;
00093   virtual Document* GetMutableDocumentPtr(int doc_id) = 0;
00095   virtual const DocTagsCollection* GetDocumentTagsPtr(int doc_id) const = 0;
00096
00098   virtual DocumentsMutableIterator AddDocument(const Document& doc) = 0;
00100   virtual void SetDocument(int doc_id, const Document& doc) = 0;
00102   virtual void RemoveDocument(int doc_id) = 0;
00104   virtual DocumentsMutableIterator MutableDocumentsBegin() = 0;
00106   virtual DocumentsMutableIterator MutableDocumentsEnd() = 0;
00107
00109   virtual DocumentsSliceIterator DocumentsSlice(const char* tag) const = 0;
00110
00112   virtual DocumentsMutableSliceIterator MutableDocumentsSlice(
00113       const char* tag) = 0;
00114
00116   virtual bool CanBuildPDFABuffer() const = 0;
00117
00119   virtual void BuildPDFABuffer() = 0;
00120
00122   virtual void GetPDFABuffer(unsigned char* output_buf, unsigned long long buf_size) const = 0;
00123
00125   virtual int GetPDFABufferSize() const = 0;
00126
00128   virtual void SetAddTextMode(const char* mode_name) = 0;
00129
00131   virtual const char* GetAddTextMode() const = 0;
00132
00134   virtual bool HasAddTextMode(const char* mode_name) const = 0;
00135
00137   virtual se::common::StringsVectorIterator AddTextModesBegin() const = 0;
00139   virtual se::common::StringsVectorIterator AddTextModesEnd() const = 0;
00140
00142   virtual void SetTextTypeMode(const char* mode_name) = 0;
00143
00145   virtual const char* GetTextTypeMode() const = 0;
00146
00148   virtual bool HasTextTypeMode(const char* mode_name) const = 0;
00149
00151   virtual se::common::StringsVectorIterator TextTypeModesBegin() const = 0;
00153   virtual se::common::StringsVectorIterator TextTypeModesEnd() const = 0;
00154
```

```
00156   virtual void SetColourMode(const bool with_colour) = 0;
00157
00159   virtual bool GetColourMode() const = 0;
00160 };
00161
00162 } } // namespace se::doc
00163
00164 #endif // DOCENGINE_DOC_RESULT_H_INCLUDED
```

2.41 doc_scene_info.h File Reference

Smart Document Engine information abbout processed scenes.

Classes

- class se::doc::DocSceneInfo

  The class representing basic information about a scene.

2.41.1 Detailed Description

Smart Document Engine information abbout processed scenes.

Definition in file doc_scene_info.h.

2.42 doc_scene_info.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_SCENE_INFO_H_INCLUDED
00012 #define DOCENGINE_DOC_SCENE_INFO_H_INCLUDED
00013
00014 #include <secommon/se_common.h>
00015 #include <docengine/doc_fields_iterators.h>
00016
00017 namespace se { namespace doc {
00018
00022 class SE_DLL_EXPORT DocSceneInfo {
00023
00024 public:
00025
00027   enum class SceneOriginType {
00028     UNDEFINED = -1,
00029     DIGITAL_BORN = 0,
00030     OPTICAL_SCANNER = 1,
00031     OPTICAL_CAMERA = 2
00032   };
00033
00035   virtual ~DocSceneInfo() = default;
00036
00038   virtual bool IsGarbage() const = 0;
00039
00041   virtual int SceneID() const = 0;
00042
00044   virtual int GarbageReasonsCount() const = 0;
00045
00047   virtual const char* GarbageReason(int idx) const = 0;
00048
00050   virtual SceneOriginType GetSceneOriginType() const = 0;
00051
00052
00054
00056   virtual int GetForensicCheckFieldsCount() const = 0;
00058   virtual bool HasForensicCheckField(const char* name) const = 0;
```

```
00060   virtual const DocForensicCheckField& GetForensicCheckField(const char* name) const = 0;
00062   virtual const DocForensicCheckField* GetForensicCheckFieldPtr(const char* name) const = 0;
00064   virtual DocForensicCheckFieldsIterator ForensicCheckFieldsBegin() const = 0;
00066   virtual DocForensicCheckFieldsIterator ForensicCheckFieldsEnd() const = 0;
00067
00068 };
00069
00070
00071
00072 } } // namespace se::doc
00073
00074 #endif // DOCENGINE_DOC_SCENE_INFO_H_INCLUDED
```

2.43 doc_session.h File Reference

Smart Document Engine image processing session.

Classes

- class se::doc::DocSession

  The class representing image processing session - main processing class of Smart Document Engine.

2.43.1 Detailed Description

Smart Document Engine image processing session.

Definition in file doc_session.h.

2.44 doc_session.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_SESSION_H_INCLUDED
00012 #define DOCENGINE_DOC_SESSION_H_INCLUDED
00013
00014 #include <secommon/se_common.h>
00015 #include <docengine/doc_forward_declarations.h>
00016
00017 namespace se { namespace doc {
00018
00019
00024 class SE_DLL_EXPORT DocSession {
00025 public:
00027   virtual ~DocSession() = default;
00028
00034   virtual DocProcessingSettings* CreateProcessingSettings() const = 0;
00035
00040   virtual const char* GetActivationRequest() = 0;
00041
00046   virtual void Activate(const char* activation_response) = 0;
00047
00052   virtual bool IsActivated() const = 0;
00053
00059   virtual void ProcessImage(const se::common::Image& in_image,
00060                             const DocProcessingSettings* settings = nullptr) = 0;
00061
00063   virtual void Reset() = 0;
00064
00066   virtual const DocResult& GetCurrentResult() const = 0;
00067
00069   virtual const DocResult* GetCurrentResultPtr() const = 0;
00070
00072   virtual const char* GetType() const = 0;
```

```
00073
00074
00075 public:
00079
00085   virtual int RegisterImage(const se::common::Image& in_image) = 0;
00086
00088   virtual void Process(DocProcessingSettings& settings) = 0;
00089
00091   virtual DocResult& GetMutableCurrentResult() = 0;
00092
00094   virtual DocResult* GetMutableCurrentResultPtr() = 0;
00095 };
00096
00097
00098 } } // namespace se::doc
00099
00100 #endif // DOCENGINE_DOC_SESSION_H_INCLUDED
```

2.45 doc_session_settings.h File Reference

Smart Document Engine session settings.

Classes

- class se::doc::DocSessionSettings

  The class representing the document processing session settings.

2.45.1 Detailed Description

Smart Document Engine session settings.

Definition in file doc_session_settings.h.

2.46 doc_session_settings.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_SESSION_SETTINGS_H_INCLUDED
00012 #define DOCENGINE_DOC_SESSION_SETTINGS_H_INCLUDED
00013
00014 #include <secommon/se_common.h>
00015 #include <docengine/doc_document_info.h>
00016
00017
00018 namespace se { namespace doc {
00019
00020
00024 class SE_DLL_EXPORT DocSessionSettings {
00025 public:
00027   virtual ~DocSessionSettings() = default;
00028
00034   virtual DocSessionSettings* Clone() const = 0;
00035
00037   virtual int GetOptionsCount() const = 0;
00039   virtual bool HasOption(const char* option_name) const = 0;
00041   virtual const char* GetOption(const char* option_name) const = 0;
00043   virtual void SetOption(const char* option_name, const char* option_value) = 0;
00045   virtual void RemoveOption(const char* option_name) = 0;
00047   virtual se::common::StringsMapIterator OptionsBegin() const = 0;
00049   virtual se::common::StringsMapIterator OptionsEnd() const = 0;
00050
00052   virtual int GetSupportedModesCount() const = 0;
00054   virtual bool HasSupportedMode(const char* mode_name) const = 0;
```

```
00056   virtual const char* GetSupportedMode(int mode_id) const = 0;
00058   virtual se::common::StringsVectorIterator SupportedModesBegin() const = 0;
00060   virtual se::common::StringsVectorIterator SupportedModesEnd() const = 0;
00061
00063   virtual const char* GetCurrentMode() const = 0;
00065   virtual void SetCurrentMode(const char* mode_name) = 0;
00066
00068   virtual int GetInternalEnginesCount() const = 0;
00071   virtual int GetSupportedDocumentTypesCount(int engine_id) const = 0;
00074   virtual bool HasSupportedDocumentType(
00075       int         engine_id,
00076       const char* doc_name) const = 0;
00079   virtual const char* GetSupportedDocumentType(
00080       int engine_id,
00081       int doc_id) const = 0;
00082
00084   virtual int GetEnabledDocumentTypesCount() const = 0;
00086   virtual bool HasEnabledDocumentType(const char* doc_name) const = 0;
00088   virtual const char* GetEnabledDocumentType(int doc_id) const = 0;
00090   virtual const DocDocumentInfo& GetDocumentInfo(const char* doc_name) const = 0;
00092   virtual const DocDocumentInfo* GetDocumentInfoPtr(const char* doc_name) const = 0;
00093
00105   virtual void AddEnabledDocumentTypes(const char* doc_type_mask) = 0;
00106
00114   virtual void RemoveEnabledDocumentTypes(const char* doc_type_mask) = 0;
00115
00118   virtual se::common::StringsSetIterator PermissiblePrefixDocMasksBegin() = 0;
00121   virtual se::common::StringsSetIterator PermissiblePrefixDocMasksEnd() = 0;
00122
00124   virtual bool IsForensicsEnabled() const = 0;
00125
00127   virtual void EnableForensics() = 0;
00128
00130   virtual void DisableForensics() = 0;
00131
00132 };
00133
00134
00135 } } // namespace se::doc
00136
00137 #endif // DOCENGINE_DOC_SESSION_SETTINGS_H_INCLUDED
```

2.47 doc_tags_collection.h File Reference

Smart Document Engine tags collection.

Classes

- class se::doc::DocTagsCollection

  The class representing the collection of tags CLASS TO BE DEPRECATED.

2.47.1 Detailed Description

Smart Document Engine tags collection.

Definition in file doc_tags_collection.h.

2.48 doc_tags_collection.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_TAGS_COLLECTION_H_INCLUDED
00012 #define DOCENGINE_DOC_TAGS_COLLECTION_H_INCLUDED
```

```
00013
00014 #include <secommon/se_common.h>
00015 #include <docengine/doc_fields_iterators.h>
00016
00017 namespace se { namespace doc {
00018
00023 class SE_DLL_EXPORT DocTagsCollection {
00024 public:
00026   virtual ~DocTagsCollection() = default;
00027
00029   virtual int GetTagsCount() const = 0;
00031   virtual bool HasTag(const char* tag) const = 0;
00033   virtual void AddTag(const char* tag) = 0;
00035   virtual void RemoveTag(const char* tag) = 0;
00036
00038   virtual se::common::StringsSetIterator TagsBegin() const = 0;
00040   virtual se::common::StringsSetIterator TagsEnd() const = 0;
00041
00043   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00044
00045 public:
00051   static DocTagsCollection* Create();
00052 };
00053
00054
00055 } } // namespace se::doc
00056
00057 #endif // DOCENGINE_DOC_TAGS_COLLECTION_H_INCLUDED
```

2.49 doc_video_session.h File Reference

Smart Document Engine video processing session.

Classes

- class se::doc::DocVideoSession

  The class representing video processing session CLASS TO BE DEPRECATED.

2.49.1 Detailed Description

Smart Document Engine video processing session.

Definition in file doc_video_session.h.

2.50 doc_video_session.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_VIDEO_SESSION_H_INCLUDED
00012 #define DOCENGINE_DOC_VIDEO_SESSION_H_INCLUDED
00013
00014 #include <secommon/se_common.h>
00015 #include <docengine/doc_forward_declarations.h>
00016
00017 namespace se { namespace doc {
00018
00019
00024 class SE_DLL_EXPORT DocVideoSession {
00025 public:
00027   virtual ~DocVideoSession() = default;
00028
00034   virtual DocProcessingSettings* CreateProcessingSettings() const = 0;
00035
```

```
00040   virtual const char* GetActivationRequest() = 0;
00041
00046   virtual void Activate(const char* activation_response) = 0;
00047
00052   virtual bool IsActivated() const = 0;
00053
00059   virtual void ProcessImage(
00060       const se::common::Image&     in_image,
00061       const DocProcessingSettings& settings) = 0;
00062
00064   virtual void Reset() = 0;
00065
00067   virtual const DocResult& GetCurrentResult() const = 0;
00069   virtual DocResult& GetMutableCurrentResult() = 0;
00070
00072   virtual const DocResult* GetCurrentResultPtr() const = 0;
00074   virtual DocResult* GetMutableCurrentResultPtr() = 0;
00075 };
00076
00077
00078 } } // namespace se::doc
00079
00080 #endif // DOCENGINE_DOC_VIDEO_SESSION_H_INCLUDED
```

2.51 doc_view.h File Reference

Smart Document Engine image view.

Classes

- class se::doc::DocView

  The class representing an image view stored in the graphical structure CLASS TO BE DEPRECATED.

2.51.1 Detailed Description

Smart Document Engine image view.

Definition in file doc_view.h.

2.52 doc_view.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_VIEW_H_INCLUDED
00012 #define DOCENGINE_DOC_VIEW_H_INCLUDED
00013
00014 #include <secommon/se_common.h>
00015 #include <docengine/doc_forward_declarations.h>
00016
00017 namespace se { namespace doc {
00018
00019
00024 class SE_DLL_EXPORT DocView {
00025 public:
00027   static const char* BaseClassNameStatic();
00028
00029 public:
00031   virtual ~DocView() = default;
00032
00034   virtual const se::common::Image& GetImage() const = 0;
00036   virtual se::common::Image& GetMutableImage() = 0;
00038   virtual const se::common::Image* GetImagePtr() const = 0;
00040   virtual se::common::Image* GetMutableImagePtr() = 0;
```

```
00042   virtual void SetImage(const se::common::Image& image) = 0;
00043
00045   virtual int GetAncestorID() const = 0;
00047   virtual void SetAncestorID(int anc_id) = 0;
00048
00050   virtual int GetRootAncestorID() const = 0;
00052   virtual void SetRootAncestorID(int root_anc_id) = 0;
00053
00056   virtual const se::common::ProjectiveTransform& GetTransform() const = 0;
00059   virtual se::common::ProjectiveTransform& GetMutableTransform() = 0;
00061   virtual void SetTransform(const se::common::ProjectiveTransform& transform) = 0;
00062
00065   virtual const se::common::ProjectiveTransform* GetTransformPtr() const = 0;
00068   virtual se::common::ProjectiveTransform* GetMutableTransformPtr() = 0;
00069
00071   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00072 };
00073
00074
00075 } } // namespace se::doc
00076
00077 #endif // DOCENGINE_DOC_VIEW_H_INCLUDED
```

2.53 doc_views_collection.h File Reference

Smart Document Engine views collection.

Classes

- class se::doc::DocViewsCollection

  The class representing the collection of views CLASS TO BE DEPRECATED.

2.53.1 Detailed Description

Smart Document Engine views collection.

Definition in file doc_views_collection.h.

2.54 doc_views_collection.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_VIEWS_COLLECTION_H_INCLUDED
00012 #define DOCENGINE_DOC_VIEWS_COLLECTION_H_INCLUDED
00013
00014 #include <secommon/se_common.h>
00015 #include <docengine/doc_forward_declarations.h>
00016 #include <docengine/doc_views_iterator.h>
00017
00018 namespace se { namespace doc {
00019
00020
00025 class SE_DLL_EXPORT DocViewsCollection {
00026 public:
00028   static const char* BaseClassNameStatic();
00029
00030 public:
00032   virtual ~DocViewsCollection() = default;
00033
00035   virtual int GetViewsCount() const = 0;
00037   virtual bool HasView(int view_id) const = 0;
00039   virtual const DocView& GetView(int view_id) const = 0;
00041   virtual DocView& GetMutableView(int view_id) = 0;
```

```
00043   virtual const DocTagsCollection& GetViewTags(int view_id) const = 0;
00045   virtual const DocView* GetViewPtr(int view_id) const = 0;
00047   virtual DocView* GetMutableViewPtr(int view_id) = 0;
00049   virtual const DocTagsCollection* GetViewTagsPtr(int view_id) const = 0;
00050
00056   virtual DocViewsMutableIterator RegisterView(
00057       const se::common::Image& image) = 0;
00058
00067   virtual DocViewsMutableIterator RegisterDerivedView(
00068       const se::common::Image&              image,
00069       int                                   ancestor_id,
00070       const se::common::ProjectiveTransform& transform) = 0;
00071
00073   virtual void DeleteOrphans() = 0;
00075   virtual void DeleteView(int view_id) = 0;
00076
00078   virtual DocViewsIterator ViewsBegin() const = 0;
00080   virtual DocViewsIterator ViewsEnd() const = 0;
00081
00083   virtual DocViewsMutableIterator MutableViewsBegin() = 0;
00085   virtual DocViewsMutableIterator MutableViewsEnd() = 0;
00086
00088   virtual DocViewsSliceIterator ViewsSlice(const char* tag) const = 0;
00089
00091   virtual DocViewsMutableSliceIterator MutableViewsSlice(const char* tag) = 0;
00092
00094   virtual void Serialize(se::common::Serializer& serializer) const = 0;
00095 };
00096
00097
00098 } } // namespace se::doc
00099
00100 #endif // DOCENGINE_DOC_VIEWS_COLLECTION_H_INCLUDED
```

2.55 doc_views_iterator.h File Reference

Smart Document Engine views iterator.

Classes

- class se::doc::DocViewsIterator

  Basic const-ref iterator for a collection of views CLASS TO BE DEPRECATED.
- class se::doc::DocViewsMutableIterator

  Mutable-ref iterator for a collection of views.
- class se::doc::DocViewsSliceIterator

  Const-ref iterator for views with a given tag.
- class se::doc::DocViewsMutableSliceIterator

  Mutable-ref iterator for views with a given tag.

2.55.1 Detailed Description

Smart Document Engine views iterator.

Definition in file doc_views_iterator.h.

2.56 doc_views_iterator.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef DOCENGINE_DOC_VIEWS_ITERATOR_H_INCLUDED
00012 #define DOCENGINE_DOC_VIEWS_ITERATOR_H_INCLUDED
00013
00014 #include <secommon/se_export_defs.h>
00015 #include <docengine/doc_forward_declarations.h>
00016
00017 namespace se { namespace doc {
00018
00019
00022 class DocViewsIteratorImpl;
00023
00028 class SE_DLL_EXPORT DocViewsIterator {
00029 private:
00031   DocViewsIterator(const DocViewsIteratorImpl& pimpl);
00032
00033 public:
00035   DocViewsIterator(const DocViewsIterator& other);
00037   DocViewsIterator& operator =(const DocViewsIterator& other);
00039   ~DocViewsIterator();
00040
00042   static DocViewsIterator ConstructFromImpl(const DocViewsIteratorImpl& pimpl);
00043
00045   int GetID() const;
00047   const DocView& GetView() const;
00049   const DocTagsCollection& GetTags() const;
00051   const DocView* GetViewPtr() const;
00053   const DocTagsCollection* GetTagsPtr() const;
00055   void Advance();
00056
00058   bool Equals(const DocViewsIterator& rvalue) const;
00060   bool operator ==(const DocViewsIterator& rvalue) const;
00062   bool operator !=(const DocViewsIterator& rvalue) const;
00063
00064 private:
00066   DocViewsIteratorImpl* pimpl_;
00067 };
00068
00069
00072 class DocViewsMutableIteratorImpl;
00073
00077 class SE_DLL_EXPORT DocViewsMutableIterator {
00078 private:
00080   DocViewsMutableIterator(const DocViewsMutableIteratorImpl& pimpl);
00081
00082 public:
00084   DocViewsMutableIterator(const DocViewsMutableIterator& other);
00086   DocViewsMutableIterator& operator =(const DocViewsMutableIterator& other);
00088   ~DocViewsMutableIterator();
00089
00091   static DocViewsMutableIterator ConstructFromImpl(
00092       const DocViewsMutableIteratorImpl& pimpl);
00093
00095   int GetID() const;
00097   const DocView& GetView() const;
00099   DocView& GetMutableView() const;
00101   const DocTagsCollection& GetTags() const;
00103   const DocView* GetViewPtr() const;
00105   DocView* GetMutableViewPtr() const;
00107   const DocTagsCollection* GetTagsPtr() const;
00109   void Advance();
00110
00112   bool Equals(const DocViewsMutableIterator& rvalue) const;
00114   bool operator ==(const DocViewsMutableIterator& rvalue) const;
00116   bool operator !=(const DocViewsMutableIterator& rvalue) const;
00117
00118 private:
00120   DocViewsMutableIteratorImpl* pimpl_;
00121 };
00122
00123
00126 class DocViewsSliceIteratorImpl;
00127
00131 class SE_DLL_EXPORT DocViewsSliceIterator {
00132 private:
00134   DocViewsSliceIterator(const DocViewsSliceIteratorImpl& pimpl);
00135
00136 public:
00138   DocViewsSliceIterator(const DocViewsSliceIterator& other);
```

```
00140   DocViewsSliceIterator& operator =(const DocViewsSliceIterator& other);
00142   ~DocViewsSliceIterator();
00143
00145   static DocViewsSliceIterator ConstructFromImpl(
00146       const DocViewsSliceIteratorImpl& pimpl);
00147
00149   int GetID() const;
00151   const DocView& GetView() const;
00153   const DocTagsCollection& GetTags() const;
00155   const DocView* GetViewPtr() const;
00157   const DocTagsCollection* GetTagsPtr() const;
00159   void Advance();
00160
00163   bool Finished() const;
00164
00165 private:
00167   DocViewsSliceIteratorImpl* pimpl_;
00168 };
00169
00170
00173 class DocViewsMutableSliceIteratorImpl;
00174
00178 class SE_DLL_EXPORT DocViewsMutableSliceIterator {
00179 private:
00181   DocViewsMutableSliceIterator(const DocViewsMutableSliceIteratorImpl& pimpl);
00182
00183 public:
00185   DocViewsMutableSliceIterator(const DocViewsMutableSliceIterator& other);
00187   DocViewsMutableSliceIterator& operator =(
00188       const DocViewsMutableSliceIterator& other);
00190   ~DocViewsMutableSliceIterator();
00191
00193   static DocViewsMutableSliceIterator ConstructFromImpl(
00194       const DocViewsMutableSliceIteratorImpl& pimpl);
00195
00197   int GetID() const;
00199   const DocView& GetView() const;
00201   DocView& GetMutableView() const;
00203   const DocTagsCollection& GetTags() const;
00205   const DocView* GetViewPtr() const;
00207   DocView* GetMutableViewPtr() const;
00209   const DocTagsCollection* GetTagsPtr() const;
00211   void Advance();
00212
00215   bool Finished() const;
00216
00217 private:
00219   DocViewsMutableSliceIteratorImpl* pimpl_;
00220 };
00221
00222
00223 } } // namespace se::doc
00224
00225 #endif // DOCENGINE_DOC_VIEWS_ITERATOR_H_INCLUDED
```

2.57 se_common.h File Reference

Include all interface headers of secommon library.

2.57.1 Detailed Description

Include all interface headers of secommon library.

Definition in file se_common.h.

2.58 se_common.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
```

```
00005
00012 #ifndef SECOMMON_SE_COMMON_H_INCLUDED
00013 #define SECOMMON_SE_COMMON_H_INCLUDED
00014
00015 #include <secommon/se_export_defs.h>
00016 #include <secommon/se_serialization.h>
00017 #include <secommon/se_string.h>
00018 #include <secommon/se_strings_iterator.h>
00019 #include <secommon/se_strings_set.h>
00020 #include <secommon/se_exception.h>
00021 #include <secommon/se_geometry.h>
00022 #include <secommon/se_image.h>
00023
00024 #endif // SECOMMON_SE_COMMON_H_INCLUDED
```

2.59 se_exception.h File Reference

Exception classes for secommon library.

Classes

- class se::common::BaseException

  BaseException class - base class for all SE exeptions. Cannot be created directly.
- class se::common::InvalidKeyException

  InvalidKeyException: thrown if to an associative container the access is performed with an invalid or a non-existent key, or if the access to a list is performed with an invalid or out-of-range index.
- class se::common::NotSupportedException

  NotSupportedException: thrown when trying to access a method which given the current state or given the passed arguments is not supported in the current version of the library or is not supported at all by design.
- class se::common::FileSystemException

  FileSystemException: thrown if an attempt is made to read from a non-existent file, or other file-system related IO error.
- class se::common::UninitializedObjectException

  UninitializedObjectException: thrown if an attempt is made to access a non-existent or non-initialized object.
- class se::common::InvalidArgumentException

  InvalidArgumentException: thrown if a method is called with invalid input parameters.
- class se::common::MemoryException

  MemoryException: thrown if an allocation is attempted with insufficient RAM.
- class se::common::InvalidStateException

  InvalidStateException: thrown if an error occurs within the system in relation to an incorrect internal state of the system objects.
- class se::common::InternalException

  InternalException: thrown if an unknown error occurs or if the error occurs within internal system components.

2.59.1 Detailed Description

Exception classes for secommon library.

Definition in file se_exception.h.

2.60 se_exception.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef SECOMMON_SE_EXCEPTION_H_INCLUDED
00012 #define SECOMMON_SE_EXCEPTION_H_INCLUDED
00013
00014 #include <secommon/se_export_defs.h>
00015
00016 namespace se { namespace common {
00017
00022 class SE_DLL_EXPORT BaseException {
00023 public:
00025   virtual ~BaseException();
00026
00028   BaseException(const BaseException& copy);
00029
00031   virtual const char* ExceptionName() const;
00032
00034   virtual const char* what() const;
00035
00036 protected:
00038   BaseException(const char* msg);
00039
00040 private:
00041   char* msg_;
00042 };
00043
00044
00050 class SE_DLL_EXPORT InvalidKeyException : public BaseException {
00051 public:
00053   InvalidKeyException(const char* msg);
00054
00056   InvalidKeyException(const InvalidKeyException& copy);
00057
00059   virtual ~InvalidKeyException() override = default;
00060
00062   virtual const char* ExceptionName() const override;
00063 };
00064
00065
00072 class SE_DLL_EXPORT NotSupportedException : public BaseException {
00073 public:
00075   NotSupportedException(const char* msg);
00076
00078   NotSupportedException(const NotSupportedException& copy);
00079
00081   virtual ~NotSupportedException() override = default;
00082
00084   virtual const char* ExceptionName() const override;
00085 };
00086
00087
00092 class SE_DLL_EXPORT FileSystemException : public BaseException {
00093 public:
00095   FileSystemException(const char* msg);
00096
00098   FileSystemException(const FileSystemException& copy);
00099
00101   virtual ~FileSystemException() override = default;
00102
00104   virtual const char* ExceptionName() const override;
00105 };
00106
00107
00112 class SE_DLL_EXPORT UninitializedObjectException : public BaseException {
00113 public:
00115   UninitializedObjectException(const char* msg);
00116
00118   UninitializedObjectException(const UninitializedObjectException& copy);
00119
00121   virtual ~UninitializedObjectException() override = default;
00122
00124   virtual const char* ExceptionName() const override;
00125 };
00126
00127
00132 class SE_DLL_EXPORT InvalidArgumentException : public BaseException {
00133 public:
00135   InvalidArgumentException(const char* msg);
00136
00138   InvalidArgumentException(const InvalidArgumentException& copy);
```

```
00139
00141   virtual ~InvalidArgumentException() override = default;
00142
00144   virtual const char* ExceptionName() const override;
00145 };
00146
00147
00152 class SE_DLL_EXPORT MemoryException : public BaseException {
00153 public:
00155   MemoryException(const char* msg);
00156
00158   MemoryException(const MemoryException& copy);
00159
00161   virtual ~MemoryException() override = default;
00162
00164   virtual const char* ExceptionName() const override;
00165 };
00166
00167
00172 class SE_DLL_EXPORT InvalidStateException : public BaseException {
00173 public:
00175   InvalidStateException(const char* msg);
00176
00178   InvalidStateException(const InvalidStateException& copy);
00179
00181   virtual ~InvalidStateException() override = default;
00182
00184   virtual const char* ExceptionName() const override;
00185 };
00186
00187
00192 class SE_DLL_EXPORT InternalException : public BaseException {
00193 public:
00195   InternalException(const char* msg);
00196
00198   InternalException(const InternalException& copy);
00199
00201   virtual ~InternalException() override = default;
00202
00204   virtual const char* ExceptionName() const override;
00205 };
00206
00207
00208 } } // namespace se::common
00209
00210 #endif // SECOMMON_SE_EXCEPTION_H_INCLUDED
```

2.61 se_export_defs.h File Reference

Export-related definitions for secommon library.

2.61.1 Detailed Description

Export-related definitions for secommon library.

Definition in file se_export_defs.h.

2.61.2 Macro Definition Documentation

SE_DLL_EXPORT

```
#define SE_DLL_EXPORT
```

Definition at line 20 of file se_export_defs.h.

2.62 se_export_defs.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef SECOMMON_SE_EXPORT_DEFS_H_INCLUDED
00012 #define SECOMMON_SE_EXPORT_DEFS_H_INCLUDED
00013
00014 #if defined _WIN32 && SE_EXPORTS
00015 # define SE_DLL_EXPORT __declspec(dllexport)
00016 #else // defined _WIN32 && SE_EXPORTS
00017 # if defined(__clang__) || defined(__GNUC__)
00018 #  define SE_DLL_EXPORT __attribute__ ((visibility ("default")))
00019 # else // clang of gnuc
00020 #  define SE_DLL_EXPORT
00021 # endif // clang of gnuc
00022 #endif // defined _WIN32 && SE_EXPORTS
00023
00024 #endif // SECOMMON_SE_EXPORT_DEFS_H_INCLUDED
```

2.63 se_geometry.h File Reference

Basic geometric classes and procedures for secommon library.

Classes

- class se::common::Rectangle

  Class representing a rectangle in an image.
- class se::common::Point

  Class representing a point in an image.
- class se::common::Size

  Class representing a size of the (rectangular) object.
- class se::common::Quadrangle

  Class representing a quadrangle in an image.
- class se::common::QuadranglesVectorIterator

  QuadranglesVectorIterator: iterator object for vector of quadrangles.
- class se::common::QuadranglesMapIterator

  QuadranglesMapIterator: iterator object for maps of named quadrangles.
- class se::common::RectanglesVectorIterator
- class se::common::Polygon

  Class representing a polygon in an image.
- class se::common::ProjectiveTransform

  Class representing projective transformation of a plane.

2.63.1 Detailed Description

Basic geometric classes and procedures for secommon library.

Definition in file se_geometry.h.

2.64 se_geometry.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef SECOMMON_SE_GEOMETRY_H_INCLUDED
00012 #define SECOMMON_SE_GEOMETRY_H_INCLUDED
00013
00014 #include <secommon/se_export_defs.h>
00015 #include <secommon/se_serialization.h>
00016
00017 namespace se { namespace common {
00018
00022 class SE_DLL_EXPORT Rectangle {
00023 public:
00025   Rectangle();
00026
00028   Rectangle(int x, int y, int width, int height);
00029
00031   void Serialize(Serializer& serializer) const;
00032
00034   void SerializeImpl(SerializerImplBase& serializer_impl) const;
00035
00036 public:
00037   int x;
00038   int y;
00039   int width;
00040   int height;
00041 };
00042
00043
00047 class SE_DLL_EXPORT Point {
00048 public:
00050   Point();
00051
00053   Point(double x, double y);
00054
00056   void Serialize(Serializer& serializer) const;
00057
00059   void SerializeImpl(SerializerImplBase& serializer_impl) const;
00060
00061 public:
00062   double x;
00063   double y;
00064 };
00065
00066
00070 class SE_DLL_EXPORT Size {
00071 public:
00073   Size();
00074
00076   Size(int width, int height);
00077
00079   void Serialize(Serializer& serializer) const;
00080
00082   void SerializeImpl(SerializerImplBase& serializer_impl) const;
00083
00084 public:
00085   int width;
00086   int height;
00087 };
00088
00089
00093 class SE_DLL_EXPORT Quadrangle {
00094 public:
00096   Quadrangle();
00097
00099   Quadrangle(const Point& a, const Point& b, const Point& c, const Point& d);
00100
00102   Point& operator[](int index);
00103
00105   const Point& operator[](int index) const;
00106
00108   const Point& GetPoint(int index) const;
00109
00111   Point& GetMutablePoint(int index);
00112
00114   void SetPoint(int index, const Point& p);
00115
00117   Rectangle GetBoundingRectangle() const;
00118
00120   void Serialize(Serializer& serializer) const;
00121
```

```
00123   void SerializeImpl(SerializerImplBase& serializer_impl) const;
00124
00125 private:
00126   Point pts_[4];
00127 };
00128
00130 class QuadranglesVectorIteratorImpl;
00131
00132
00136 class SE_DLL_EXPORT QuadranglesVectorIterator {
00137 private:
00139   QuadranglesVectorIterator(const QuadranglesVectorIteratorImpl& pimpl);
00140
00141 public:
00143   QuadranglesVectorIterator(const QuadranglesVectorIterator& other);
00144
00146   QuadranglesVectorIterator& operator =(const QuadranglesVectorIterator& other);
00147
00149   ~QuadranglesVectorIterator();
00150
00152   static QuadranglesVectorIterator ConstructFromImpl(
00153     const QuadranglesVectorIteratorImpl& pimpl);
00154
00156   const Quadrangle& GetValue() const;
00157
00159   bool Equals(const QuadranglesVectorIterator& rvalue) const;
00160
00162   bool operator ==(const QuadranglesVectorIterator& rvalue) const;
00163
00165   bool operator !=(const QuadranglesVectorIterator& rvalue) const;
00166
00168   void Advance();
00169
00171   void operator ++();
00172
00173 private:
00174   class QuadranglesVectorIteratorImpl* pimpl_;
00175 };
00176
00177
00179 class QuadranglesMapIteratorImpl;
00180
00184 class SE_DLL_EXPORT QuadranglesMapIterator {
00185 private:
00187   QuadranglesMapIterator(const QuadranglesMapIteratorImpl& pimpl);
00188
00189 public:
00191   QuadranglesMapIterator(const QuadranglesMapIterator& other);
00192
00194   QuadranglesMapIterator& operator =(const QuadranglesMapIterator& other);
00195
00197   ~QuadranglesMapIterator();
00198
00200   static QuadranglesMapIterator ConstructFromImpl(
00201       const QuadranglesMapIteratorImpl& pimpl);
00202
00204   const char* GetKey() const;
00205
00207   const Quadrangle& GetValue() const;
00208
00210   bool Equals(const QuadranglesMapIterator& rvalue) const;
00211
00213   bool operator ==(const QuadranglesMapIterator& rvalue) const;
00214
00216   bool operator !=(const QuadranglesMapIterator& rvalue) const;
00217
00219   void Advance();
00220
00222   void operator ++();
00223
00224 private:
00225   class QuadranglesMapIteratorImpl* pimpl_;
00226 };
00227
00228 class RectanglesVectorIteratorImpl;
00229
00230 class SE_DLL_EXPORT RectanglesVectorIterator {
00231 private:
00233   RectanglesVectorIterator(const RectanglesVectorIteratorImpl& pimpl);
00234
00235 public:
00237   RectanglesVectorIterator(const RectanglesVectorIterator& other);
00238
00240   RectanglesVectorIterator& operator =(const RectanglesVectorIterator& other);
00241
00243   ~RectanglesVectorIterator();
00244
```

```
00246   static RectanglesVectorIterator ConstructFromImpl(
00247       const RectanglesVectorIteratorImpl& pimpl);
00248
00250   const Rectangle& GetValue() const;
00251
00253   bool Equals(const RectanglesVectorIterator& rvalue) const;
00254
00256   bool operator ==(const RectanglesVectorIterator& rvalue) const;
00257
00259   bool operator !=(const RectanglesVectorIterator& rvalue) const;
00260
00262   void Advance();
00263
00265   void operator ++();
00266
00267 private:
00268   class RectanglesVectorIteratorImpl* pimpl_;
00269 };
00270
00274 class SE_DLL_EXPORT Polygon {
00275 public:
00277   Polygon();
00278
00280   Polygon(const Point* points, int points_count);
00281
00283   Polygon(const Polygon& other);
00284
00286   Polygon& operator =(const Polygon& other);
00287
00289   ~Polygon();
00290
00292   int GetPointsCount() const;
00293
00295   const Point* GetPoints() const;
00296
00298   Point& operator [](int index);
00299
00301   const Point& operator [](int index) const;
00302
00304   const Point& GetPoint(int index) const;
00305
00307   Point& GetMutablePoint(int index);
00308
00310   void SetPoint(int index, const Point& p);
00311
00315   void Resize(int size);
00316
00318   Rectangle GetBoundingRectangle() const;
00319
00321   void Serialize(Serializer& serializer) const;
00322
00324   void SerializeImpl(SerializerImplBase& serializer_impl) const;
00325
00326 private:
00327   int pts_cnt_;
00328   Point* pts_;
00329 };
00330
00331
00335 class SE_DLL_EXPORT ProjectiveTransform {
00336 public:
00337   using Raw2dArrayType = double[3][3];
00338
00339 public:
00340
00348   static bool CanCreate(const Quadrangle& src_quad, const Quadrangle& dst_quad);
00349
00358   static bool CanCreate(const Quadrangle& src_quad, const Size& dst_size);
00359
00364   static ProjectiveTransform* Create();
00365
00373   static ProjectiveTransform* Create(
00374       const Quadrangle& src_quad,
00375       const Quadrangle& dst_quad);
00376
00384   static ProjectiveTransform* Create(
00385       const Quadrangle& src_quad,
00386       const Size&       dst_size);
00387
00393   static ProjectiveTransform* Create(const Raw2dArrayType& coeffs);
00394
00395 public:
00397   virtual ~ProjectiveTransform() = default;
00398
00400   virtual ProjectiveTransform* Clone() const = 0;
00401
00403   virtual Point TransformPoint(const Point& p) const = 0;
```

```
00404
00406   virtual Quadrangle TransformQuad(const Quadrangle& q) const = 0;
00407
00409   virtual Polygon TransformPolygon(const Polygon& poly) const = 0;
00410
00412   virtual bool IsInvertable() const = 0;
00413
00415   virtual void Invert() = 0;
00416
00418   virtual ProjectiveTransform* CloneInverted() const = 0;
00419
00421   virtual const Raw2dArrayType& GetRawCoeffs() const = 0;
00422
00424   virtual Raw2dArrayType& GetMutableRawCoeffs() = 0;
00425
00427   virtual void Serialize(Serializer& serializer) const = 0;
00428 };
00429
00430
00431 } } // namespace se::common
00432
00433 #endif // SECOMMON_SE_GEOMETRY_H_INCLUDED
```

2.65 se_image.h File Reference

secommon library Image

Classes

- class se::common::YUVDimensions

  The YUVDimensions struct - extended YUV parameters.
- class se::common::Image

  Class representing bitmap image.

Variables

- IPF_G = 0

  Greyscale.
- IPF_GA

  Greyscale + Alpha.
- IPF_AG

  Alpha + Greyscale.
- IPF_RGB

  RGB.
- IPF_BGR

  BGR.
- IPF_BGRA

  BGR + Alpha.
- IPF_ARGB

  Alpha + RGB.
- YUVTYPE_UNDEFINED = 0

  No format.
- YUVTYPE_NV21 = 1

  NV 21.

2.65.1 Detailed Description

secommon library Image

Definition in file se_image.h.

2.65.2 Variable Documentation

IPF_G

```
IPF_G = 0
```

Greyscale.

Definition at line 27 of file se_image.h.

IPF_GA

```
IPF_GA
```

Greyscale + Alpha.

Definition at line 28 of file se_image.h.

IPF_AG

```
IPF_AG
```

Alpha + Greyscale.

Definition at line 29 of file se_image.h.

IPF_RGB

```
IPF_RGB
```

RGB.

Definition at line 30 of file se_image.h.

IPF_BGR

```
IPF_BGR
```

BGR.

Definition at line 31 of file se_image.h.

IPF_BGRA

```
IPF_BGRA
```

BGR + Alpha.

Definition at line 32 of file se_image.h.

IPF_ARGB

```
IPF_ARGB
```

Alpha + RGB.

Definition at line 33 of file se_image.h.

YUVTYPE_UNDEFINED

```
YUVTYPE_UNDEFINED = 0
```

No format.

Definition at line 41 of file se_image.h.

YUVTYPE_NV21

```
YUVTYPE_NV21 = 1
```

NV 21.

Definition at line 42 of file se_image.h.

2.66 se_image.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef SECOMMON_SE_IMAGE_H_INCLUDED
00012 #define SECOMMON_SE_IMAGE_H_INCLUDED
00013
00014 #include <secommon/se_export_defs.h>
00015 #include <secommon/se_geometry.h>
00016 #include <secommon/se_serialization.h>
00017 #include <secommon/se_string.h>
00018
00019 #include <secommon/se_images_iterator.h>
00020
00021 namespace se { namespace common {
00022
00026 enum SE_DLL_EXPORT ImagePixelFormat {
00027   IPF_G = 0,
00028   IPF_GA,
00029   IPF_AG,
00030   IPF_RGB,
00031   IPF_BGR,
00032   IPF_BGRA,
```

```
00033   IPF_ARGB,
00034   IPF_RGBA
00035 };
00036
00040 enum SE_DLL_EXPORT YUVType {
00041   YUVTYPE_UNDEFINED = 0,
00042   YUVTYPE_NV21 = 1,
00043   YUVTYPE_420_888 = 2
00044 };
00045
00049 class SE_DLL_EXPORT YUVDimensions {
00050 public:
00052   YUVDimensions();
00053
00055   YUVDimensions(int y_pixel_stride,
00056                 int y_row_stride,
00057                 int u_pixel_stride,
00058                 int u_row_stride,
00059                 int v_pixel_stride,
00060                 int v_row_stride,
00061                 int width,
00062                 int height,
00063                 YUVType type);
00064
00065   int y_plane_pixel_stride;
00066   int y_plane_row_stride;
00067   int u_plane_pixel_stride;
00068   int u_plane_row_stride;
00069   int v_plane_pixel_stride;
00070   int v_plane_row_stride;
00071   int width;
00072   int height;
00073   YUVType type;
00074 };
00075
00079 class SE_DLL_EXPORT Image {
00080 public:
00086   static int GetNumberOfPages(const char* image_filename);
00087
00094   static MutableString GetImagePageName(const char *image_filename,
00095                                         int page_number);
00096
00102   static Image* CreateEmpty();
00103
00113   static Image* FromFile(
00114       const char* image_filename,
00115       const int   page_number = 0,
00116       const Size& max_size = Size(25000, 25000));
00117
00128   static Image* FromFileBuffer(
00129       unsigned char* data,
00130       int            data_length,
00131       const int      page_number = 0,
00132       const Size&    max_size = Size(25000, 25000));
00133
00147   static Image* FromBuffer(
00148       unsigned char* raw_data,
00149       int            raw_data_length,
00150       int            width,
00151       int            height,
00152       int            stride,
00153       int            channels);
00154
00168   static Image* FromBufferExtended(
00169       unsigned char*   raw_data,
00170       int              raw_data_length,
00171       int              width,
00172       int              height,
00173       int              stride,
00174       ImagePixelFormat pixel_format,
00175       int              bytes_per_channel);
00176
00186   static Image* FromYUVBuffer(
00187       unsigned char* yuv_data,
00188       int            yuv_data_length,
00189       int            width,
00190       int            height);
00191
00192
00205   static Image* FromYUV(
00206       unsigned char*       y_plane,
00207       int                  y_plane_length,
00208       unsigned char*       u_plane,
00209       int                  u_plane_length,
00210       unsigned char*       v_plane,
00211       int                  v_plane_length,
00212       const YUVDimensions& dimensions);
```

```
00213
00223   static Image* FromBase64Buffer(
00224       const char* base64_buffer,
00225       const int   page_number = 0,
00226       const Size& max_size = Size(25000, 25000));
00227
00228 public:
00230   virtual ~Image() = default;
00231
00236   virtual int GetNumberOfLayers() const = 0;
00237
00243   virtual const Image& GetLayer(const char* name) const = 0;
00244
00250   virtual const Image* GetLayerPtr(const char* name) const = 0;
00251
00256   virtual ImagesMapIterator LayersBegin() const = 0;
00257
00262   virtual ImagesMapIterator LayersEnd() const = 0;
00263
00269   virtual bool HasLayer(const char* name) const = 0;
00270
00275   virtual bool HasLayers() const = 0;
00276
00281   virtual void RemoveLayer(const char* name) = 0;
00282
00284   virtual void RemoveLayers() = 0;
00285
00292   virtual void SetLayer(const char* name, const Image& image) = 0;
00293
00301   virtual void SetLayerWithOwnership(const char* name, Image* image) = 0;
00302
00303 public:
00309   virtual Image* CloneDeep() const = 0;
00310
00318   virtual Image* CloneShallow() const = 0;
00319
00321   virtual void Clear() = 0;
00322
00328   virtual int GetRequiredBufferLength() const = 0;
00329
00337   virtual int CopyToBuffer(unsigned char* buffer, int buffer_length) const = 0;
00338
00339 #ifndef STRICT_DATA_CONTAINMENT
00345   virtual void Save(const char* image_filename) const = 0;
00346 #endif // #ifndef STRICT_DATA_CONTAINMENT
00347
00353   virtual int GetRequiredBase64BufferLength() const = 0;
00354
00363   virtual int CopyBase64ToBuffer(
00364       char* out_buffer, int buffer_length) const = 0;
00365
00370   virtual MutableString GetBase64String() const = 0;
00371
00377   virtual double EstimateFocusScore(double quantile = 0.95) const = 0;
00378
00383   virtual void Resize(const Size& new_size) = 0;
00384
00391   virtual Image* CloneResized(const Size& new_size) const = 0;
00392
00399   virtual void Crop(const Quadrangle& quad) = 0;
00400
00408   virtual Image* CloneCropped(const Quadrangle& quad) const = 0;
00409
00415   virtual void Crop(const Quadrangle& quad, const Size& size) = 0;
00416
00424   virtual Image* CloneCropped(const Quadrangle& quad, const Size& size) const = 0;
00425
00430   virtual void Crop(const Rectangle& rect) = 0;
00431
00439   virtual Image* CloneCropped(const Rectangle& rect) const = 0;
00440
00450   virtual Image* CloneCroppedShallow(const Rectangle& rect) const = 0;
00451
00458   virtual void Mask(const Rectangle& rect, int pixel_expand = 0, double pixel_density = 0) = 0;
00459
00467   virtual Image* CloneMasked(const Rectangle& rect, int pixel_expand = 0) const = 0;
00468
00474   virtual void Mask(const Quadrangle& quad, int pixel_expand = 0, double pixel_density = 0) = 0;
00475
00484   virtual Image* CloneMasked(const Quadrangle& quad, int pixel_expand = 0) const = 0;
00485
00496   virtual void Fill(const Rectangle& rect, int ch1, int ch2 = 0, int ch3 = 0, int ch4 = 0, int
      pixel_expand = 0) = 0;
00497
00510   virtual Image* CloneFilled(const Rectangle& rect, int ch1, int ch2 = 0, int ch3 = 0, int ch4 = 0,
      int pixel_expand = 0) const = 0;
00511
```

```
00522   virtual void Fill(const Quadrangle& quad, int ch1, int ch2 = 0, int ch3 = 0, int ch4 = 0, int
      pixel_expand = 0) = 0;
00523
00536   virtual Image* CloneFilled(const Quadrangle& quad, int ch1, int ch2 = 0, int ch3 = 0, int ch4 = 0,
      int pixel_expand = 0) const = 0;
00537
00541   virtual void FlipVertical() = 0;
00542
00548   virtual Image* CloneFlippedVertical() const = 0;
00549
00553   virtual void FlipHorizontal() = 0;
00554
00560   virtual Image* CloneFlippedHorizontal() const = 0;
00561
00566   virtual void Rotate90(int times) = 0;
00567
00574   virtual Image* CloneRotated90(int times) const = 0;
00575
00579   virtual void AverageChannels() = 0;
00580
00586   virtual Image* CloneAveragedChannels() const = 0;
00587
00591   virtual void Invert() = 0;
00592
00598   virtual Image* CloneInverted() const = 0;
00599
00601   virtual int GetWidth() const = 0;
00602
00604   virtual int GetHeight() const = 0;
00605
00607   virtual Size GetSize() const = 0;
00608
00610   virtual int GetStride() const = 0;
00611
00613   virtual int GetChannels() const = 0;
00614
00616   virtual void* GetUnsafeBufferPtr() const = 0;
00617
00619   virtual bool IsMemoryOwner() const = 0;
00620
00622   virtual void ForceMemoryOwner() = 0;
00623
00625   virtual void Serialize(Serializer& serializer) const = 0;
00626 };
00627
00628
00629 } } // namespace se::common
00630
00631 #endif // SECOMMON_SE_IMAGE_H_INCLUDED
```

2.67 se_serialization.h File Reference

Facilities for serialization of objects.

Classes

- class se::common::SerializationParameters

  Class representing serialization parameters.
- class se::common::Serializer

  Class representing the serializer object.

2.67.1 Detailed Description

Facilities for serialization of objects.

Definition in file se_serialization.h.

2.68 se_serialization.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef SECOMMON_SE_SERIALIZATION_H_INCLUDED
00012 #define SECOMMON_SE_SERIALIZATION_H_INCLUDED
00013
00014 #include <secommon/se_export_defs.h>
00015 #include <secommon/se_strings_iterator.h>
00016
00017 namespace se { namespace common {
00018
00020 class SerializationParametersImpl;
00021
00025 class SE_DLL_EXPORT SerializationParameters {
00026 public:
00028   SerializationParameters();
00030   ~SerializationParameters();
00032   SerializationParameters(const SerializationParameters& copy);
00034   SerializationParameters& operator =(
00035       const SerializationParameters& other);
00036
00037 public:
00044   bool HasIgnoredObjectType(const char* object_type) const;
00045
00050   void AddIgnoredObjectType(const char* object_type);
00051
00056   void RemoveIgnoredObjectType(const char* object_type);
00057
00059   se::common::StringsSetIterator IgnoredObjectTypesBegin() const;
00060
00062   se::common::StringsSetIterator IgnoredObjectTypesEnd() const;
00063
00069   bool HasIgnoredKey(const char* key) const;
00070
00075   void AddIgnoredKey(const char* key);
00076
00081   void RemoveIgnoredKey(const char* key);
00082
00084   se::common::StringsSetIterator IgnoredKeysBegin() const;
00085
00087   se::common::StringsSetIterator IgnoredKeysEnd() const;
00088
00089 public:
00091   const SerializationParametersImpl& GetImpl() const;
00092
00093 private:
00094   SerializationParametersImpl* pimpl_;
00095 };
00096
00097
00099 class SerializerImplBase;
00100
00104 class SE_DLL_EXPORT Serializer {
00105 public:
00107   virtual ~Serializer() = default;
00108
00110   virtual void Reset() = 0;
00111
00113   virtual const char* GetCStr() const = 0;
00114
00116   virtual const char* SerializerType() const = 0;
00117
00118 public:
00125   static Serializer* CreateJSONSerializer(
00126       const SerializationParameters& params);
00127 };
00128
00129
00130 } } // namespace se::common
00131
00132 #endif // SECOMMON_SE_SERIALIZATION_H_INCLUDED
```

2.69 se_string.h File Reference

OcrString and related classes for secommon library.

Classes

- class se::common::MutableString

  Class representing a mutable, memory-owner string.
- class se::common::OcrCharVariant

  Class representing a possible character recognition result.
- class se::common::OcrPair
- class se::common::OcrChar

  Class representing an OCR information for a given recognized character.
- class se::common::OcrString

  Class representing text string recognition result.
- class se::common::ByteString

  Class representing byte string.

2.69.1 Detailed Description

OcrString and related classes for secommon library.

Definition in file se_string.h.

2.70 se_string.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef SECOMMON_SE_STRING_H_INCLUDED
00012 #define SECOMMON_SE_STRING_H_INCLUDED
00013
00014 #include <cstddef>
00015 #include <cstdint>
00016 #include <secommon/se_export_defs.h>
00017 #include <secommon/se_geometry.h>
00018 #include <secommon/se_serialization.h>
00019
00020 namespace se { namespace common {
00021
00025 class SE_DLL_EXPORT MutableString {
00026 public:
00028   MutableString();
00029
00031   explicit MutableString(const char* c_str);
00032
00034   MutableString(const MutableString& other);
00035
00037   MutableString& operator =(const MutableString& other);
00038
00040   ~MutableString();
00041
00043   MutableString& operator +=(const MutableString& other);
00044
00046   MutableString operator +(const MutableString& other) const;
00047
00049   const char* GetCStr() const;
00050
00053   int GetLength() const;
00054
00056   void Serialize(Serializer& serializer) const;
00057
00059   void SerializeImpl(SerializerImplBase& serializer_impl) const;
00060
00061 private:
00062   int len_;
00063   char* buf_;
00064 };
00065
```

```
00066
00070 class SE_DLL_EXPORT OcrCharVariant {
00071 public:
00073   OcrCharVariant();
00074
00080   OcrCharVariant(const MutableString& utf8_char, float confidence);
00081
00087   OcrCharVariant(const char* utf8_char, float confidence);
00088
00090   ~OcrCharVariant() = default;
00091
00093   const char* GetCharacter() const;
00094
00096   void SetCharacter(const MutableString& utf8_char);
00097
00099   void SetCharacter(const char* utf8_char);
00100
00102   float GetConfidence() const;
00103
00105   void SetConfidence(float confidence);
00106
00108   float GetInternalScore() const;
00109
00111   void SetInternalScore(float internal_score);
00112
00114   void Serialize(Serializer& serializer) const;
00115
00117   void SerializeImpl(SerializerImplBase& serializer_impl) const;
00118
00119 private:
00120   MutableString char_;
00121   float conf_;
00122   float internal_score_;
00123 };
00124
00125
00126
00127 class SE_DLL_EXPORT OcrPair {
00128 public:
00130   OcrPair();
00131
00137   OcrPair(const MutableString& utf8_char, float confidence);
00138
00144   OcrPair(const char* utf8_char, float confidence);
00145
00151   OcrPair(const MutableString& utf8_char, int confidence);
00152
00158   OcrPair(const char* utf8_char, int confidence);
00159
00161   ~OcrPair() = default;
00162
00164   const char* GetCharacter() const;
00165
00167   void SetCharacter(const MutableString& utf8_char);
00168
00170   void SetCharacter(const char* utf8_char);
00171
00173   unsigned char GetConfidence() const;
00174
00176   void SetConfidence(float confidence);
00177
00179   void SetConfidence(int confidence);
00180
00182   void Serialize(Serializer& serializer) const;
00183
00185   void SerializeImpl(SerializerImplBase& serializer_impl) const;
00186
00187 private:
00188   MutableString char_;
00189   unsigned char conf_;
00190 };
00191
00192
00196 class SE_DLL_EXPORT OcrChar {
00197 public:
00199   OcrChar();
00200
00208   OcrChar(const OcrCharVariant* variants,
00209           int                   variants_count,
00210           bool                  is_highlighted,
00211           const Quadrangle&     quad);
00212
00214   OcrChar(const OcrChar& other);
00215
00217   OcrChar& operator =(const OcrChar& other);
00218
00220   ~OcrChar();
```

```
00221
00223   int GetVariantsCount() const;
00224
00226   const OcrCharVariant* GetVariants() const;
00227
00229   OcrCharVariant& operator [](int index);
00230
00232   const OcrCharVariant& operator [](int index) const;
00233
00235   const OcrCharVariant& GetVariant(int index) const;
00236
00238   OcrCharVariant& GetMutableVariant(int index);
00239
00241   void SetVariant(int index, const OcrCharVariant& v);
00242
00244   void Resize(int size);
00245
00247   bool GetIsHighlighted() const;
00248
00250   void SetIsHighlighted(bool is_highlighted);
00251
00253   const Quadrangle& GetQuadrangle() const;
00254
00256   Quadrangle& GetMutableQuadrangle();
00257
00259   void SetQuadrangle(const Quadrangle& quad);
00260
00262   void SortVariants();
00263
00265   const OcrCharVariant& GetFirstVariant() const;
00266
00268   void Serialize(Serializer& serializer) const;
00269
00271   void SerializeImpl(SerializerImplBase& serializer_impl) const;
00272
00273 private:
00274   int vars_cnt_;
00275   OcrCharVariant* vars_;
00276   bool is_highlighted_;
00277   Quadrangle quad_;
00278 };
00279
00280
00282 class OcrStringImpl;
00283
00287 class SE_DLL_EXPORT OcrString {
00288 private:
00290   OcrString(const OcrStringImpl& ocr_string_impl);
00291
00292 public:
00294   OcrString();
00295
00301   OcrString(const char* utf8_str);
00302
00308   OcrString(const OcrChar* chars, int chars_count);
00309
00311   OcrString(const OcrString& other);
00312
00314   OcrString& operator =(const OcrString& other);
00315
00317   ~OcrString();
00318
00323   static OcrString ConstructFromImpl(const class OcrStringImpl& ocr_string_impl);
00324
00326   const class OcrStringImpl* GetOcrStringImplPtr() const;
00327
00329   int GetCharsCount() const;
00330
00332   const OcrChar* GetChars() const;
00333
00335   OcrChar& operator [](int index);
00336
00338   const OcrChar& operator [](int index) const;
00339
00341   const OcrChar& GetChar(int index) const;
00342
00344   OcrChar& GetMutableChar(int index);
00345
00347   void SetChar(int index, const OcrChar& chr);
00348
00350   void AppendChar(const OcrChar& chr);
00351
00353   void AppendString(const OcrString& str);
00354
00356   void Resize(int size);
00357
00359   const Quadrangle GetQuadrangleByIndex(int idx) const;
```

```
00360
00362   bool GetAcceptFlag() const;
00363
00365   bool GetCellAcceptFlagByIndex(int idx) const;
00366
00368   float GetBestVariantConfidenceByIndex(int idx) const;
00369
00371   void SortVariants();
00372
00374   MutableString GetFirstString() const;
00375
00377   void UnpackChars();
00378
00380   void RepackChars();
00381
00383   void Serialize(Serializer& serializer) const;
00384
00386   void SerializeImpl(SerializerImplBase& serializer_impl) const;
00387
00389   OcrPair* GetPairTable();
00390 private:
00391   OcrStringImpl* ocr_string_impl_;
00392 };
00393
00397 class SE_DLL_EXPORT ByteString {
00398 public:
00400   ByteString();
00401
00403   ~ByteString();
00404
00406   explicit ByteString(const unsigned char* bytes, size_t n);
00407
00409   ByteString(const ByteString &other);
00410
00412   ByteString &operator=(const ByteString &other);
00413
00415   void swap(ByteString &other) noexcept;
00416
00418   int GetLength() const noexcept;
00419
00421   int GetRequiredBase64BufferLength() const;
00422
00424   int CopyBase64ToBuffer(char* out_buffer, int buffer_length) const;
00425
00427   MutableString GetBase64String() const;
00428
00430   int GetRequiredHexBufferLength() const;
00431
00433   int CopyHexToBuffer(char* out_buffer, int buffer_length) const;
00434
00436   MutableString GetHexString() const;
00437
00438 private:
00439   size_t len_;
00440   uint8_t *buf_;
00441 };
00442
00443 } } // namespace se::common::
00444
00445 #endif // SECOMMON_SE_STRING_H_INCLUDED
```

2.71 se_strings_iterator.h File Reference

String iterators used in SE libraries.

Classes

- class se::common::StringsVectorIterator

  Iterator to a vector-like collection of strings.
- class se::common::StringsSetIterator

  Iterator to a set-like collection of strings.
- class se::common::StringsMapIterator

  Iterator to a map from strings to strings.

2.71.1 Detailed Description

String iterators used in SE libraries.

Definition in file se_strings_iterator.h.

2.72 se_strings_iterator.h

Go to the documentation of this file.

```
00001 /*
00002   Copyright (c) 2016-2025, Smart Engines Service LLC
00003   All rights reserved.
00004 */
00005
00011 #ifndef SECOMMON_SE_STRINGS_ITERATOR_H_INCLUDED
00012 #define SECOMMON_SE_STRINGS_ITERATOR_H_INCLUDED
00013
00014 #include <secommon/se_export_defs.h>
00015
00016 namespace se { namespace common {
00017
00018
00020 class StringsVectorIteratorImpl;
00021
00022
00026 class SE_DLL_EXPORT StringsVectorIterator {
00027 private:
00029   StringsVectorIterator(const StringsVectorIteratorImpl& pimpl);
00030
00031 public:
00033   StringsVectorIterator(const StringsVectorIterator& other);
00034
00036   StringsVectorIterator& operator =(const StringsVectorIterator& other);
00037
00039   ~StringsVectorIterator();
00040
00042   static StringsVectorIterator ConstructFromImpl(
00043       const StringsVectorIteratorImpl& pimpl);
00044
00046   const char* GetValue() const;
00047
00049   bool Equals(const StringsVectorIterator& rvalue) const;
00050
00052   bool operator ==(const StringsVectorIterator& rvalue) const;
00053
00055   bool operator !=(const StringsVectorIterator& rvalue) const;
00056
00058   void Advance();
00059
00061   void operator ++();
00062
00063 private:
00064   class StringsVectorIteratorImpl* pimpl_;
00065 };
00066
00067
00069 class StringsSetIteratorImpl;
00070
00071
00075 class SE_DLL_EXPORT StringsSetIterator {
00076 private:
00078   StringsSetIterator(const StringsSetIteratorImpl& pimpl);
00079
00080 public:
00082   StringsSetIterator(const StringsSetIterator& other);
00083
00085   StringsSetIterator& operator =(const StringsSetIterator& other);
00086
00088   ~StringsSetIterator();
00089
00091   static StringsSetIterator ConstructFromImpl(
00092       const StringsSetIteratorImpl& pimpl);
00093
00095   const char* GetValue() const;
00096
00098   bool Equals(const StringsSetIterator& rvalue) const;
00099
00101   bool operator ==(const StringsSetIterator& rvalue) const;
00102
```

```
00104   bool operator !=(const StringsSetIterator& rvalue) const;
00105
00107   void Advance();
00108
00110   void operator ++();
00111
00112 private:
00113   class StringsSetIteratorImpl* pimpl_;
00114 };
00115
00116
00118 class StringsMapIteratorImpl;
00119
00120
00124 class SE_DLL_EXPORT StringsMapIterator {
00125 private:
00127   StringsMapIterator(const StringsMapIteratorImpl& pimpl);
00128
00129 public:
00131   StringsMapIterator(const StringsMapIterator& other);
00132
00134   StringsMapIterator& operator =(const StringsMapIterator& other);
00135
00137   ~StringsMapIterator();
00138
00140   static StringsMapIterator ConstructFromImpl(
00141       const StringsMapIteratorImpl& pimpl);
00142
00144   const char* GetKey() const;
00145
00147   const char* GetValue() const;
00148
00150   bool Equals(const StringsMapIterator& rvalue) const;
00151
00153   bool operator==(const StringsMapIterator& rvalue) const;
00154
00156   bool operator!=(const StringsMapIterator& rvalue) const;
00157
00159   void Advance();
00160
00162   void operator ++();
00163
00164 private:
00165   class StringsMapIteratorImpl* pimpl_;
00166 };
00167
00168
00169 } } // namespace se::common::
00170
00171 #endif // SECOMMON_SE_STRINGS_ITERATOR_H_INCLUDED
```

Index