

Scribo

User documentation 2.1

Generated by Doxygen 1.8.1.2

Tue Jul 8 2014 17:16:02

Contents

1	Documentation of Scribo	1
1.1	Introduction	1
1.2	Quickstart	1
1.3	Copyright and License.	3
2	Module Index	5
2.1	Modules	5
3	Namespace Index	7
3.1	Namespace List	7
4	Class Index	9
4.1	Class Hierarchy	9
5	Class Index	13
5.1	Class List	13
6	Module Documentation	17
6.1	Binarization	17
6.1.1	Detailed Description	17
6.2	Kim	18
6.2.1	Detailed Description	18
6.2.2	Function Documentation	18
6.2.2.1	kim	18
6.2.2.2	kim	18
6.3	Niblack	19
6.3.1	Detailed Description	19
6.3.2	Function Documentation	19
6.3.2.1	niblack	19
6.3.2.2	niblack	20
6.3.2.3	niblack	20
6.3.2.4	niblack_fast	20
6.3.2.5	niblack_fast	20

6.3.2.6	niblack_fast	20
6.3.2.7	niblack_threshold	21
6.3.2.8	niblack_threshold	21
6.3.2.9	niblack_threshold	21
6.4	Otsu	22
6.4.1	Detailed Description	22
6.4.2	Function Documentation	22
6.4.2.1	otsu	22
6.4.2.2	otsu_threshold	22
6.5	Sauvola	23
6.5.1	Detailed Description	23
6.5.2	comparison	24
6.5.3	Function Documentation	24
6.5.3.1	sauvola	24
6.5.3.2	sauvola	25
6.5.3.3	sauvola	25
6.5.3.4	sauvola_ms	25
6.5.3.5	sauvola_ms	26
6.5.3.6	sauvola_ms	26
6.5.3.7	sauvola_ms_split	26
6.5.3.8	sauvola_ms_split	26
6.5.3.9	sauvola_ms_split	26
6.5.3.10	sauvola_threshold	26
6.5.3.11	sauvola_threshold	27
6.5.3.12	sauvola_threshold	27
6.6	Wolf	28
6.6.1	Detailed Description	28
6.6.2	Function Documentation	28
6.6.2.1	wolf	28
6.6.2.2	wolf	28
6.6.2.3	wolf	29
6.6.2.4	wolf_fast	29
6.6.2.5	wolf_fast	29
6.6.2.6	wolf_fast	29
6.7	Data structures	30
6.7.1	Detailed Description	30
6.8	Debug	31
6.8.1	Detailed Description	33
6.8.2	Function Documentation	33
6.8.2.1	alignment_decision_image	33

6.8.2.2	bboxes_enlarged_image	33
6.8.2.3	bboxes_enlarged_image	34
6.8.2.4	bboxes_image	34
6.8.2.5	bboxes_image	34
6.8.2.6	bboxes_image	34
6.8.2.7	bboxes_image	34
6.8.2.8	char_space_image	34
6.8.2.9	decision_image	34
6.8.2.10	decision_image	35
6.8.2.11	highlight_text_area	35
6.8.2.12	highlight_text_area	35
6.8.2.13	highlight_text_area	35
6.8.2.14	line_info_image	36
6.8.2.15	linked_bboxes_image	36
6.8.2.16	linked_bboxes_image	36
6.8.2.17	linked_bboxes_image	36
6.8.2.18	linked_bboxes_image	37
6.8.2.19	linked_bboxes_image	37
6.8.2.20	links_decision_image	37
6.8.2.21	links_image	37
6.8.2.22	logger	37
6.8.2.23	looks_like_a_text_line_image	38
6.8.2.24	looks_like_a_text_line_image	38
6.8.2.25	mean_and_base_lines_image	38
6.8.2.26	mean_and_base_lines_image	38
6.8.2.27	save_comp_diff	39
6.8.2.28	save_label_image	39
6.8.2.29	save_table_image	39
6.8.2.30	save_table_image	39
6.8.2.31	text_areas_image	39
6.8.2.32	text_color_image	39
6.9	Component Group Filtering	40
6.9.1	Detailed Description	40
6.9.2	Function Documentation	40
6.9.2.1	object_groups_mean_width	40
6.9.2.2	object_groups_size_ratio	40
6.9.2.3	object_groups_small	40
6.9.2.4	object_groups_with_holes	41
6.10	Component Link Filtering	42
6.10.1	Detailed Description	42

6.10.2	Function Documentation	42
6.10.2.1	object_links_aligned	42
6.10.2.2	object_links_bbox_h_ratio	43
6.10.2.3	object_links_bbox_overlap	43
6.10.2.4	object_links_bbox_ratio	43
6.10.2.5	object_links_bbox_w_ratio	44
6.10.2.6	object_links_bottom_aligned	44
6.10.2.7	object_links_center_aligned	44
6.10.2.8	object_links_left_aligned	45
6.10.2.9	object_links_non_aligned_simple	45
6.10.2.10	object_links_right_aligned	46
6.10.2.11	object_links_top_aligned	46
6.11	Component Filtering	48
6.11.1	Detailed Description	49
6.11.2	Function Documentation	49
6.11.2.1	components_large	49
6.11.2.2	components_large	49
6.11.2.3	components_on_border	49
6.11.2.4	components_small	50
6.11.2.5	components_small	50
6.11.2.6	components_thin	50
6.11.2.7	components_thin	51
6.11.2.8	components_v_thin	51
6.11.2.9	components_with_two_holes	51
6.11.2.10	objects_h_thick	51
6.11.2.11	objects_h_thick	52
6.11.2.12	objects_h_thin	52
6.11.2.13	objects_h_thin	52
6.11.2.14	objects_size_ratio	52
6.11.2.15	objects_thick	53
6.11.2.16	objects_thick	53
6.11.2.17	objects_v_thick	53
6.11.2.18	objects_v_thick	54
6.11.2.19	objects_v_thin	54
6.11.2.20	objects_with_holes	54
6.12	Element Filtering	55
6.12.1	Detailed Description	55
6.12.2	Function Documentation	55
6.12.2.1	images_in_paragraph	55
6.12.2.2	objects_in_borders	55

6.12.2.3	separators_in_borders	56
6.12.2.4	separators_in_element	56
6.12.2.5	separators_in_paragraph	56
6.12.2.6	separators_vert_in_borders	57
6.13	Element Filtering	58
6.13.1	Detailed Description	58
6.14	Line Link Filtering	59
6.14.1	Detailed Description	59
6.14.2	Function Documentation	59
6.14.2.1	line_links_x_height	59
6.15	Paragraph Filtering	60
6.15.1	Detailed Description	60
6.15.2	Function Documentation	60
6.15.2.1	paragraphs_bbox_overlap	60
6.15.2.2	paragraphs_in_borders	60
6.15.2.3	paragraphs_in_image	60
6.16	Layout Analysis	62
6.16.1	Detailed Description	62
6.16.2	Function Documentation	62
6.16.2.1	xy_cut	62
6.17	Components Extraction	63
6.17.1	Detailed Description	63
6.17.2	Function Documentation	63
6.17.2.1	components	63
6.17.2.2	components	63
6.18	Primitive Extraction	65
6.18.1	Detailed Description	65
6.19	Lines and Separators extraction	66
6.19.1	Detailed Description	67
6.19.2	Function Documentation	67
6.19.2.1	horizontal_separators	67
6.19.2.2	lines_discontinued	67
6.19.2.3	lines_h_discontinued	68
6.19.2.4	lines_h_pattern	68
6.19.2.5	lines_h_single	69
6.19.2.6	lines_h_single	69
6.19.2.7	lines_h_thick_and_single	69
6.19.2.8	lines_h_thick_and_thin	70
6.19.2.9	lines_pattern	70
6.19.2.10	lines_thick	70

6.19.2.11	lines_thick	70
6.19.2.12	lines_v_discontinued	71
6.19.2.13	lines_v_pattern	71
6.19.2.14	lines_v_single	71
6.19.2.15	lines_v_single	72
6.19.2.16	lines_v_thick_and_single	72
6.19.2.17	separators	72
6.19.2.18	separators_nonvisible	73
6.19.2.19	vertical_separators	73
6.20	Routines	74
6.20.1	Detailed Description	74
6.21	Text Recognition	75
6.21.1	Detailed Description	75
6.21.2	Function Documentation	75
6.21.2.1	recognition	75
6.21.2.2	recognition	75
6.22	Text Extraction	76
6.22.1	Detailed Description	77
6.22.2	Function Documentation	77
6.22.2.1	extract_lines	77
6.22.2.2	extract_lines	77
6.22.2.3	extract_lines	77
6.22.2.4	extract_lines_with_features	77
6.22.2.5	extract_lines_with_features	78
6.22.2.6	extract_lines_wo_merge	78
6.22.2.7	extract_lines_wo_merge	78
6.22.2.8	extract_lines_wo_merge	78
6.22.2.9	extract_lines_wo_merge	78
6.22.2.10	extract_paragraphs	78
6.22.2.11	extract_paragraphs_hdoc	79
6.22.2.12	link_lines	79
6.22.2.13	merging	79
6.22.2.14	merging_hdoc	79
6.23	Toolchains	80
6.23.1	Detailed Description	80
6.23.2	Function Documentation	80
6.23.2.1	text_extraction	80
6.24	Pictures	81
6.24.1	Detailed Description	81
6.25	Documents	82

6.25.1 Detailed Description	82
6.26 Preprocessing	83
6.27 Preprocessing	84
6.27.1 Detailed Description	84
6.27.2 Function Documentation	84
6.27.2.1 text_in_doc_preprocess	84
6.27.2.2 text_in_doc_preprocess	84
6.27.2.3 text_in_doc_preprocess	85
6.27.2.4 text_in_doc_preprocess	85
6.27.2.5 text_in_doc_preprocess	85
6.28 Processing	86
6.28.1 Detailed Description	86
6.28.2 Function Documentation	86
6.28.2.1 text_in_picture	86
6.29 Processing	87
6.29.1 Detailed Description	87
6.29.2 Function Documentation	87
6.29.2.1 content_in_doc	87
6.29.2.2 content_in_hdoc	88
6.29.2.3 text_in_doc	88
7 Namespace Documentation	89
7.1 scribo Namespace Reference	89
7.1.1 Detailed Description	91
7.1.2 Typedef Documentation	91
7.1.2.1 component_id_t	91
7.1.3 Function Documentation	91
7.1.3.1 central_sites	91
7.1.3.2 erase_objects	92
7.2 scribo::binarization Namespace Reference	92
7.2.1 Detailed Description	94
7.2.2 Function Documentation	94
7.2.2.1 global_threshold	94
7.2.2.2 global_threshold_auto	95
7.2.2.3 local_threshold	95
7.2.2.4 sauvola_ms	95
7.2.2.5 sauvola_ms	95
7.2.2.6 singh	95
7.2.2.7 singh	96
7.2.2.8 singh	96

7.3	scribo::component Namespace Reference	96
7.3.1	Detailed Description	96
7.3.2	Enumeration Type Documentation	97
7.3.2.1	Tag	97
7.3.2.2	Type	97
7.3.3	Function Documentation	97
7.3.3.1	operator<<	97
7.3.3.2	operator<<	97
7.3.3.3	str2tag	98
7.3.3.4	str2type	98
7.4	scribo::core Namespace Reference	98
7.4.1	Detailed Description	98
7.5	scribo::debug Namespace Reference	98
7.5.1	Detailed Description	100
7.5.2	Enumeration Type Documentation	101
7.5.2.1	Level	101
7.5.2.2	VerboseMode	101
7.5.3	Function Documentation	101
7.5.3.1	txt_to_verbose_mode	101
7.5.3.2	usage	101
7.6	scribo::draw Namespace Reference	101
7.6.1	Detailed Description	102
7.6.2	Function Documentation	102
7.6.2.1	bounding_box_links	102
7.6.2.2	bounding_box_links	102
7.6.2.3	bounding_box_links	102
7.6.2.4	bounding_box_links	103
7.6.2.5	bounding_boxes	103
7.6.2.6	bounding_boxes	103
7.6.2.7	groups_bboxes	103
7.7	scribo::filter Namespace Reference	103
7.7.1	Detailed Description	105
7.8	scribo::make Namespace Reference	105
7.8.1	Detailed Description	106
7.8.2	Function Documentation	106
7.8.2.1	debug_filename	106
7.8.2.2	influence_zone_graph	106
7.8.2.3	paragraph	106
7.8.2.4	paragraph	106
7.8.2.5	paragraph	106

7.8.2.6	text_blocks_image	107
7.8.2.7	text_components_image	107
7.9	scribo::postprocessing Namespace Reference	107
7.9.1	Detailed Description	107
7.9.2	Function Documentation	107
7.9.2.1	fill_object_holes	107
7.9.2.2	images_to_drop_capital	107
7.10	scribo::preprocessing Namespace Reference	107
7.10.1	Detailed Description	108
7.10.2	Function Documentation	108
7.10.2.1	crop	108
7.10.2.2	crop_without_localization	109
7.10.2.3	denoise	109
7.10.2.4	denoise_bg	109
7.10.2.5	denoise_fg	110
7.10.2.6	deskew	110
7.10.2.7	homogeneous_contrast	110
7.10.2.8	homogeneous_contrast	110
7.10.2.9	rotate_90	111
7.10.2.10	rotate_90	111
7.10.2.11	split_bg_fg	111
7.11	scribo::primitive Namespace Reference	111
7.11.1	Detailed Description	112
7.12	scribo::primitive::extract Namespace Reference	112
7.12.1	Detailed Description	113
7.12.2	Function Documentation	114
7.12.2.1	alignments	114
7.12.2.2	canvas	115
7.12.2.3	cells	115
7.12.2.4	non_text	115
7.12.2.5	non_text_hdoc	116
7.13	scribo::primitive::group Namespace Reference	116
7.13.1	Detailed Description	116
7.13.2	Function Documentation	116
7.13.2.1	apply	116
7.13.2.2	apply	116
7.13.2.3	from_double_link	117
7.13.2.4	from_double_link_any	117
7.13.2.5	from_graph	117
7.13.2.6	from_single_link	117

7.14 scribo::primitive::internal Namespace Reference	118
7.14.1 Detailed Description	118
7.14.2 Function Documentation	118
7.14.2.1 find_graph_link	118
7.14.2.2 find_link	118
7.14.2.3 find_root	119
7.14.2.4 have_link_valid	119
7.14.2.5 is_link_valid	119
7.14.2.6 rd	119
7.14.2.7 update_graph_link	119
7.15 scribo::primitive::link Namespace Reference	120
7.15.1 Detailed Description	122
7.15.2 Function Documentation	122
7.15.2.1 compute	122
7.15.2.2 compute	123
7.15.2.3 compute_several	123
7.15.2.4 merge_double_link	123
7.15.2.5 merge_double_link_closest_aligned	123
7.15.2.6 with_graph	124
7.15.2.7 with_rag	124
7.15.2.8 with_several_graphes	124
7.15.2.9 with_several_left_links	124
7.15.2.10 with_several_left_links	124
7.15.2.11 with_several_right_closest_links	124
7.15.2.12 with_several_right_closest_links	125
7.15.2.13 with_several_right_links	125
7.15.2.14 with_several_right_links	125
7.15.2.15 with_several_right_links_overlap	125
7.15.2.16 with_several_right_links_overlap	125
7.15.2.17 with_single_down_link	125
7.15.2.18 with_single_down_link	126
7.15.2.19 with_single_down_link	126
7.15.2.20 with_single_left_link	126
7.15.2.21 with_single_left_link	126
7.15.2.22 with_single_left_link_dmax_ratio	126
7.15.2.23 with_single_left_link_dmax_ratio	127
7.15.2.24 with_single_left_link_dmax_ratio	127
7.15.2.25 with_single_left_link_dmax_ratio	127
7.15.2.26 with_single_left_link_dmax_ratio_aligned	127
7.15.2.27 with_single_left_link_dmax_ratio_aligned	127

7.15.2.28	with_single_left_link_dmax_ratio_aligned	128
7.15.2.29	with_single_right_link	128
7.15.2.30	with_single_right_link	128
7.15.2.31	with_single_right_link_bottom	128
7.15.2.32	with_single_right_link_bottom	128
7.15.2.33	with_single_right_link_dmax_ratio	129
7.15.2.34	with_single_right_link_dmax_ratio	129
7.15.2.35	with_single_right_link_dmax_ratio	129
7.15.2.36	with_single_right_link_dmax_ratio	129
7.15.2.37	with_single_right_link_dmax_ratio_aligned	129
7.15.2.38	with_single_right_link_dmax_ratio_aligned	130
7.15.2.39	with_single_right_link_dmax_ratio_aligned	130
7.15.2.40	with_single_right_link_top	130
7.15.2.41	with_single_right_link_top	130
7.15.2.42	with_single_up_link	130
7.15.2.43	with_single_up_link	131
7.15.2.44	with_single_up_link	131
7.16	scribo::table Namespace Reference	131
7.16.1	Detailed Description	132
7.16.2	Function Documentation	132
7.16.2.1	align_lines_horizontally	132
7.16.2.2	align_lines_vertically	132
7.16.2.3	connect_horizontal_lines	132
7.16.2.4	connect_vertical_lines	133
7.16.2.5	erase	133
7.16.2.6	extract	133
7.16.2.7	rebuild	133
7.16.2.8	repair_horizontal_lines	134
7.16.2.9	repair_vertical_lines	134
7.17	scribo::table::internal Namespace Reference	134
7.17.1	Detailed Description	135
7.17.2	Function Documentation	135
7.17.2.1	align_lines	135
7.17.2.2	connect_lines	135
7.17.2.3	repair_lines	135
7.18	scribo::text Namespace Reference	135
7.18.1	Detailed Description	137
7.18.2	Function Documentation	137
7.18.2.1	clean	137
7.18.2.2	clean_inplace	137

7.18.2.3	look_like_text_lines	137
7.18.2.4	look_like_text_lines_inplace	137
8	Class Documentation	139
8.1	cluster_stats< T > Class Template Reference	139
8.1.1	Detailed Description	139
8.2	compare_values< T > Struct Template Reference	139
8.2.1	Detailed Description	139
8.3	mIn::info Struct Reference	140
8.3.1	Detailed Description	140
8.4	scribo::binarization::internal::niblack_formula Struct Reference	140
8.4.1	Detailed Description	140
8.4.2	Member Function Documentation	140
8.4.2.1	operator()	140
8.4.2.2	operator()	141
8.5	scribo::binarization::internal::niblack_functor< I > Struct Template Reference	141
8.5.1	Detailed Description	141
8.6	scribo::binarization::internal::niblack_functor_fast< I > Struct Template Reference	141
8.6.1	Detailed Description	142
8.7	scribo::binarization::internal::niblack_threshold_functor< I > Struct Template Reference	142
8.7.1	Detailed Description	142
8.8	scribo::binarization::internal::sauvola_formula Struct Reference	143
8.8.1	Detailed Description	143
8.8.2	Member Function Documentation	143
8.8.2.1	operator()	143
8.8.2.2	operator()	143
8.9	scribo::binarization::internal::sauvola_functor< I > Struct Template Reference	143
8.9.1	Detailed Description	144
8.10	scribo::binarization::internal::sauvola_ms_functor< I > Struct Template Reference	144
8.10.1	Detailed Description	145
8.11	scribo::binarization::internal::sauvola_threshold_functor< I > Struct Template Reference	145
8.11.1	Detailed Description	145
8.12	scribo::binarization::internal::singh_formula< V > Struct Template Reference	145
8.12.1	Detailed Description	146
8.12.2	Member Function Documentation	146
8.12.2.1	operator()	146
8.12.2.2	operator()	146
8.13	scribo::binarization::internal::singh_functor< I > Struct Template Reference	146
8.13.1	Detailed Description	147
8.14	scribo::binarization::internal::wolf_formula< V > Struct Template Reference	147

8.14.1 Detailed Description	147
8.14.2 Member Function Documentation	147
8.14.2.1 operator()	147
8.15 scribo::binarization::internal::wolf_funcutor< I > Struct Template Reference	147
8.15.1 Detailed Description	148
8.16 scribo::binarization::internal::wolf_funcutor_fast< I > Struct Template Reference	148
8.16.1 Detailed Description	149
8.17 scribo::component_features_data Struct Reference	149
8.17.1 Detailed Description	149
8.18 scribo::component_info< L > Class Template Reference	149
8.18.1 Detailed Description	150
8.18.2 Member Function Documentation	150
8.18.2.1 accept	150
8.19 scribo::component_set< L > Class Template Reference	150
8.19.1 Detailed Description	151
8.19.2 Constructor & Destructor Documentation	152
8.19.2.1 component_set	152
8.19.2.2 component_set	152
8.19.2.3 component_set	152
8.19.2.4 component_set	153
8.19.2.5 component_set	153
8.19.3 Member Function Documentation	153
8.19.3.1 accept	153
8.19.3.2 add_separators	153
8.19.3.3 clear_separators	153
8.19.3.4 duplicate	153
8.19.3.5 has_separators	153
8.19.3.6 info	153
8.19.3.7 info	153
8.19.3.8 init_	154
8.19.3.9 is_valid	154
8.19.3.10 labeled_image	154
8.19.3.11 nelements	154
8.19.3.12 operator()	154
8.19.3.13 operator()	154
8.19.3.14 separators	154
8.19.3.15 update_tags	154
8.19.4 Friends And Related Function Documentation	154
8.19.4.1 operator<<	154
8.19.4.2 operator==	154

8.20 scribo::debug::arg_data Struct Reference	154
8.20.1 Detailed Description	155
8.21 scribo::debug::internal::logger_ Class Reference	155
8.21.1 Detailed Description	155
8.21.2 Member Function Documentation	155
8.21.2.1 default_verbose_mode	155
8.21.2.2 filename_prefix	156
8.21.2.3 is_at_level	156
8.21.2.4 is_at_verbose_mode	156
8.21.2.5 is_enabled	156
8.21.2.6 is_verbose	156
8.21.2.7 level	156
8.21.2.8 log	156
8.21.2.9 log_image	156
8.21.2.10 operator<<	156
8.21.2.11 set_default_verbose_mode	156
8.21.2.12 set_filename_prefix	156
8.21.2.13 set_level	156
8.21.2.14 set_verbose_mode	157
8.21.2.15 set_verbose_prefix	157
8.21.2.16 start_time_logging	157
8.21.2.17 stop_time_logging	157
8.21.2.18 verbose_mode	157
8.22 scribo::debug::opt_data Struct Reference	157
8.22.1 Detailed Description	157
8.23 scribo::debug::option_parser Class Reference	157
8.23.1 Detailed Description	158
8.24 scribo::debug::toggle_data Struct Reference	158
8.24.1 Detailed Description	158
8.25 scribo::DMax_Functor< E > Class Template Reference	158
8.25.1 Detailed Description	158
8.26 scribo::doc_serializer< E > Class Template Reference	158
8.26.1 Detailed Description	159
8.27 scribo::document< L > Class Template Reference	159
8.27.1 Detailed Description	160
8.27.2 Member Function Documentation	160
8.27.2.1 accept	160
8.27.2.2 has_text	160
8.28 scribo::fun::v2b::components_large_filter< L > Struct Template Reference	160
8.28.1 Detailed Description	161

8.28.2	Constructor & Destructor Documentation	161
8.28.2.1	components_large_filter	161
8.28.3	Member Function Documentation	161
8.28.3.1	operator()	161
8.28.4	Member Data Documentation	162
8.28.4.1	components_	162
8.28.4.2	marked_	162
8.28.4.3	max_size_	162
8.28.4.4	nlabels_	162
8.29	scribo::fun::v2b::components_on_border_filter< L > Struct Template Reference	162
8.29.1	Detailed Description	163
8.29.2	Constructor & Destructor Documentation	163
8.29.2.1	components_on_border_filter	163
8.29.3	Member Function Documentation	163
8.29.3.1	operator()	163
8.29.4	Member Data Documentation	163
8.29.4.1	b_	163
8.29.4.2	components_	163
8.30	scribo::fun::v2b::components_small_filter< L > Struct Template Reference	164
8.30.1	Detailed Description	164
8.30.2	Constructor & Destructor Documentation	164
8.30.2.1	components_small_filter	164
8.30.3	Member Function Documentation	164
8.30.3.1	operator()	165
8.30.4	Member Data Documentation	165
8.30.4.1	components_	165
8.30.4.2	marked_	165
8.30.4.3	min_size_	165
8.30.4.4	nlabels_	165
8.31	scribo::fun::v2b::label_to_bool< L > Struct Template Reference	165
8.31.1	Detailed Description	166
8.32	scribo::fun::v2v::highlight< R > Struct Template Reference	166
8.32.1	Detailed Description	166
8.33	scribo::group_info Class Reference	166
8.33.1	Detailed Description	166
8.34	scribo::internal::component_set_data< L > Struct Template Reference	167
8.34.1	Detailed Description	167
8.35	scribo::internal::document_data< L > Struct Template Reference	167
8.35.1	Detailed Description	168
8.36	scribo::internal::line_info_data< L > Struct Template Reference	168

8.36.1 Detailed Description	169
8.37 scribo::internal::line_links_data< L > Struct Template Reference	169
8.37.1 Detailed Description	169
8.38 scribo::internal::line_set_data< L > Struct Template Reference	169
8.38.1 Detailed Description	170
8.39 scribo::internal::object_groups_data< L > Struct Template Reference	170
8.39.1 Detailed Description	170
8.40 scribo::internal::object_links_data< L > Struct Template Reference	170
8.40.1 Detailed Description	171
8.41 scribo::internal::paragraph_set_data< L > Struct Template Reference	171
8.41.1 Detailed Description	171
8.42 scribo::internal::sort_comp_ids< L > Struct Template Reference	171
8.42.1 Detailed Description	172
8.43 scribo::io::img::internal::debug_img_visitor< L > Class Template Reference	172
8.43.1 Detailed Description	172
8.44 scribo::io::img::internal::full_img_visitor< L > Class Template Reference	172
8.44.1 Detailed Description	173
8.45 scribo::io::img::internal::non_text_img_visitor Class Reference	173
8.45.1 Detailed Description	173
8.46 scribo::io::img::internal::text_img_visitor Class Reference	173
8.46.1 Detailed Description	174
8.47 scribo::io::xml::internal::color_t Struct Reference	174
8.47.1 Detailed Description	174
8.48 scribo::io::xml::internal::extended_page_xml_visitor< L > Class Template Reference	174
8.48.1 Detailed Description	174
8.49 scribo::io::xml::internal::full_xml_visitor Class Reference	175
8.49.1 Detailed Description	175
8.50 scribo::io::xml::internal::page_xml_visitor< L > Class Template Reference	175
8.50.1 Detailed Description	176
8.51 scribo::layout::internal::hist_info Struct Reference	176
8.51.1 Detailed Description	176
8.52 scribo::layout::internal::node< B > Class Template Reference	176
8.52.1 Detailed Description	176
8.53 scribo::line_info< L > Class Template Reference	176
8.53.1 Detailed Description	178
8.53.2 Constructor & Destructor Documentation	178
8.53.2.1 line_info	178
8.53.2.2 line_info	178
8.53.2.3 line_info	178
8.53.2.4 line_info	178

8.53.3	Member Function Documentation	178
8.53.3.1	accept	178
8.53.3.2	delta_of_line	178
8.53.3.3	ebbox	178
8.53.3.4	fast_merge	178
8.53.3.5	force_stats_update	179
8.53.3.6	id	179
8.53.3.7	is_hidden	179
8.53.3.8	operator=	179
8.53.3.9	precise_merge	179
8.53.3.10	set_hidden	179
8.53.3.11	update_ebbox	179
8.54	scribo::line_links< L > Class Template Reference	179
8.54.1	Detailed Description	180
8.54.2	Member Function Documentation	180
8.54.2.1	accept	180
8.55	scribo::line_set< L > Class Template Reference	180
8.55.1	Detailed Description	181
8.55.2	Constructor & Destructor Documentation	181
8.55.2.1	line_set	181
8.55.2.2	line_set	181
8.55.2.3	line_set	181
8.55.3	Member Function Documentation	181
8.55.3.1	components	181
8.55.3.2	components_	181
8.55.3.3	compute_lines	181
8.55.3.4	duplicate	181
8.55.3.5	force_stats_update	181
8.55.3.6	groups	182
8.55.3.7	info	182
8.55.3.8	info	182
8.55.3.9	infos	182
8.55.3.10	infos_	182
8.55.3.11	is_valid	182
8.55.3.12	links	182
8.55.3.13	nelements	182
8.55.3.14	operator()	182
8.55.3.15	operator()	182
8.55.3.16	update_tags	182
8.55.3.17	update_types	183

8.56 scribo::Link_Functor< E > Class Template Reference	183
8.56.1 Detailed Description	183
8.57 scribo::object_groups< L > Class Template Reference	183
8.57.1 Detailed Description	184
8.57.2 Member Function Documentation	184
8.57.2.1 accept	184
8.57.2.2 group_of	184
8.57.2.3 operator()	184
8.58 scribo::object_links< L > Class Template Reference	184
8.58.1 Detailed Description	185
8.58.2 Constructor & Destructor Documentation	185
8.58.2.1 object_links	185
8.58.2.2 object_links	185
8.58.2.3 object_links	185
8.58.3 Member Function Documentation	185
8.58.3.1 accept	186
8.58.3.2 clear	186
8.58.3.3 comp_to_link	186
8.58.3.4 components	186
8.58.3.5 disable_linking	186
8.58.3.6 duplicate	186
8.58.3.7 has_linking_enabled	186
8.58.3.8 init	186
8.58.3.9 is_linked	186
8.58.3.10 is_valid	186
8.58.3.11 nelements	186
8.58.3.12 operator()	187
8.58.3.13 update	187
8.58.4 Friends And Related Function Documentation	187
8.58.4.1 operator<<	187
8.59 scribo::paragraph_info< L > Class Template Reference	187
8.59.1 Detailed Description	187
8.59.2 Member Function Documentation	188
8.59.2.1 add_line	188
8.60 scribo::paragraph_set< L > Class Template Reference	188
8.60.1 Detailed Description	188
8.60.2 Member Function Documentation	188
8.60.2.1 accept	188
8.61 scribo::preprocessing::internal::Hough Class Reference	189
8.61.1 Detailed Description	189

8.62	scribo::preprocessing::internal::QCompare Struct Reference	189
8.62.1	Detailed Description	189
8.63	scribo::preprocessing::internal::s_angle Struct Reference	189
8.63.1	Detailed Description	189
8.64	scribo::primitive::link::internal::dmax_default Class Reference	189
8.64.1	Detailed Description	190
8.65	scribo::primitive::link::internal::dmax_functor_base< E > Class Template Reference	190
8.65.1	Detailed Description	190
8.66	scribo::primitive::link::internal::dmax_hrules Class Reference	190
8.66.1	Detailed Description	191
8.67	scribo::primitive::link::internal::dmax_width_and_height Class Reference	191
8.67.1	Detailed Description	191
8.68	scribo::primitive::link::internal::dmax_width_only Class Reference	191
8.68.1	Detailed Description	192
8.69	scribo::primitive::link::internal::link_functor_base< L, E > Class Template Reference	192
8.69.1	Detailed Description	193
8.69.2	Constructor & Destructor Documentation	193
8.69.2.1	link_functor_base	193
8.69.3	Member Function Documentation	193
8.69.3.1	invalidate_link	193
8.69.3.2	start_point	193
8.69.3.3	validate_link	194
8.70	scribo::primitive::link::internal::link_several_dmax_base< L, E > Class Template Reference	194
8.70.1	Detailed Description	195
8.70.2	Member Function Documentation	195
8.70.2.1	invalidate_link	195
8.70.2.2	start_point	195
8.70.2.3	validate_link	195
8.71	scribo::primitive::link::internal::link_single_dmax_base< L, E > Class Template Reference	195
8.71.1	Detailed Description	196
8.71.2	Member Function Documentation	196
8.71.2.1	invalidate_link	196
8.71.2.2	start_point	197
8.71.2.3	validate_link	197
8.72	scribo::primitive::link::internal::link_single_dmax_ratio_aligned_base< L, F, E > Class Template Reference	197
8.72.1	Detailed Description	198
8.72.2	Member Function Documentation	198
8.72.2.1	invalidate_link	198
8.72.2.2	start_point	198

8.72.2.3	validate_link	198
8.73	scribo::primitive::link::internal::link_single_dmax_ratio_aligned_delta_base< L, F, E > Class Template Reference	199
8.73.1	Detailed Description	200
8.73.2	Member Function Documentation	200
8.73.2.1	invalidate_link	200
8.73.2.2	start_point	200
8.73.2.3	validate_link	200
8.74	scribo::primitive::link::internal::link_single_dmax_ratio_base< L, F, E > Class Template Reference	200
8.74.1	Detailed Description	201
8.74.2	Member Function Documentation	202
8.74.2.1	invalidate_link	202
8.74.2.2	start_point	202
8.74.2.3	validate_link	202
8.75	scribo::Serializable< E > Class Template Reference	202
8.75.1	Detailed Description	202
8.75.2	Member Function Documentation	202
8.75.2.1	accept	202
8.76	scribo::SerializeVisitor< E > Class Template Reference	203
8.76.1	Detailed Description	203
8.77	scribo::toolchain::internal::content_in_doc_functor< I > Struct Template Reference	203
8.77.1	Detailed Description	204
8.78	scribo::toolchain::internal::content_in_hdoc_functor< I > Struct Template Reference	204
8.78.1	Detailed Description	205
8.79	scribo::toolchain::internal::text_in_doc_functor< I > Struct Template Reference	205
8.79.1	Detailed Description	205
8.80	scribo::toolchain::internal::text_in_doc_preprocess_functor< I > Struct Template Reference	205
8.80.1	Detailed Description	206
8.81	scribo::toolchain::internal::text_in_picture_functor< I > Struct Template Reference	207
8.81.1	Detailed Description	207
8.82	scribo::toolchain::internal::Toolchain_Functor Class Reference	208
8.82.1	Detailed Description	208
8.83	scribo::util::integral_sub_sum_sum2_functor< I, S > Class Template Reference	208
8.83.1	Detailed Description	208
8.84	scribo::util::integral_sum_sum2_functor< V, S > Class Template Reference	209
8.84.1	Detailed Description	209
8.85	scribo::util::integral_sum_sum2_global_min_functor< V, S > Class Template Reference	209
8.85.1	Detailed Description	209
8.86	stats< T > Class Template Reference	210
8.86.1	Detailed Description	210

Chapter 1

Documentation of Scribo

1.1 Introduction

Scribo, a framework for Document Image Analysis.

Its initial release has been developed in the context of the SCRIBO project of the Free Software Thematic Group, part of the "Systematic Paris-Région" Cluster (France). This project was partially funded by the French government, its economic development agencies and by the Paris-Région institutions.

It aims at providing the following features:

- routines for DIA,
- DIA toolchains,
- text detection in document,
- document layout analysis,
- high-level data structures,
- novel algorithms and techniques,
- standard I/O,
- graphical user interfaces,
- and command Line Interfaces.

Its code is based on Milena, the generic and efficient c++ image processing library of Olena.

It is integrated in the KDE semantic desktop, Nepomuk, in the current Mandriva Linux distribution.

1.2 Quickstart

No tutorial nor quick reference guide are written for now. But a good start is to read the code of existing processing chain. You can take a look at the following files :

- [text_in_doc_preprocess.hh](#)
- [content_in_doc_functor.hh](#)
- [text_in_picture.hh](#)

Take also a look at the [Modules section](#).

BUSINESS LIFE

camp, and in such surprisingly spacious beds, that it took them hours to get to sleep. Where were you, you ask? Why, in our driveway, of course. The only suitable place to do a stop-in this case was out of the motor before really hitting the highway.

About 1 A.M. I awoke, frozen, and watched another piece of vital information flash before me during the bustle: how we work the heating system. I hustled with a flashlight and the outside gas tank and finally figured it out. The next morning, however, I learned that I had been too slow. My 3-year-old son, Willie, awoke with a sore head cold. The next day: Our destination—the dry bottom of El Mirador in north the last of the year's speed trials was shut down because of 35 mph winds.

Instead we turned the motor into a portable pack in Newport Beach, Calif. And although a questionable location, some great, usually strange, the variety of our temporary home, the gas pump nearby, and our motor plus party.

Journalism/graduate KOTR/Photo.com



THE AIR OF AIRSTREAM
LTC. with a mobile
hospital in 1991,
parked in Red Square
in 1980; and sitting
in a speed record
with a '60 Dodge.



enlightened up outdoors. We were in business mode, talking with lots of random environmental and enjoying every simple moment of it. We even forgot to use the far-south TV.

After a few days the reality of life in

the run eventually crept in (which would happen to me in anything short of a reasonable time frame), and we could not journey. I noticed that I had finally missed the red point. Airstreams are hot again because they are high-end folk art, sculpture that represent American

style and style. In an age where people at the peak end of the rising curve are starting to scale back on all that is the art world, Airstream's statements about the simple life without anything looks or combine especially when you customize them (see bar). To this point, 100% (and growing) of today's American buyers are "design oriented" who see Airstream as cool retro vehicles. They are then in awe, says, then with the aesthetic and design system to give home. (Long familiar: designer Dieter Rams. Airstream now offers an incredibly chic lifestyle.)

I just hope that Airstream can bridge all its different customers and remain faithful to the details being lost in the process. As it was with many longtime brands, the timeless have kept it alive but it is the new blood who will make or break the future. If

MY PLEA TO ALAN MULALLY

In which the author begs Ford CEO to produce the Ford Airstream.

DEAR ALAN: I am writing you because I recently had the opportunity to spend an afternoon with your advanced design team and their brilliant Ford Airstream hybrid design had not I thought that you would be there. I was once again struck by its beautiful design. Incorporation of Airstream's iconic shell, its clever solutions for convenience and cost, and its ability to provide a simple lifestyle. You may agree these capabilities are not easily put together for any

recent family member from Ford. The Ford Airstream concept already achieved something that I honestly thought would never be possible. It took one design who is essentially a minimalist. It is a car that can be a family vehicle despite the fact, imagine how easy it will be to transport homes who already own such a thing over with a simple gas engine or hybrid system. But, you must already know this. So what will you announce production? Sincerely, Sam



HIGH CONCEPT
Ford's upcoming
Airstream car



64 FORTUNE November 26, 2007

1.3 Copyright and License.

Copyright (C) 2011, 2012, 2013, 2014 EPITA Research and Development (LRDE).

This documentation is part of Olena.

Olena is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, version 2 of the License.

Olena is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with Olena. If not, see <http://www.gnu.org/licenses/>.

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Data structures	30
Routines	74
Binarization	17
Kim	18
Niblack	19
Otsu	22
Sauvola	23
Wolf	28
Components Extraction	63
Debug	31
Element Filtering	58
Component Filtering	48
Component Group Filtering	40
Component Link Filtering	42
Element Filtering	55
Line Link Filtering	59
Paragraph Filtering	60
Layout Analysis	62
Primitive Extraction	65
Lines and Separators extraction	66
Text Extraction	76
Text Recognition	75
Toolchains	80
Documents	82
Preprocessing	84
Processing	87
Pictures	81
Preprocessing	83
Processing	86

Chapter 3

Namespace Index

3.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

scribo	The main namespace of the Scribo module	89
scribo::binarization	Namespace of binarization routines	92
scribo::component	Namespace of classes related to components	96
scribo::core	Namespace of core routines	98
scribo::debug	Namespace of debug routines	98
scribo::draw	Namespace of drawing routines	101
scribo::filter	Namespace of filtering routines	103
scribo::make	Namespace of routines constructing objects	105
scribo::postprocessing	Namespace of postprocessing routines	107
scribo::preprocessing	Namespace of preprocessing routines	107
scribo::primitive	Namespace of primitive related routines	111
scribo::primitive::extract	Namespace of primitive extraction related routines	112
scribo::primitive::group	Namespace of primitive grouping related routines	116
scribo::primitive::internal	Namespace of internal routines grouping text components	118
scribo::primitive::link	Namespace of primitive linking related routines	120
scribo::table	Namespace of routines working on tables	131
scribo::table::internal	Namespace of internal routines working on tables	134
scribo::text	Namespace of routines working on text components	135

Chapter 4

Class Index

4.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

cluster_stats< T >	139
compare_values< T >	139
mln::Function_v2b< components_large_filter< L > >[external]	
scribo::fun::v2b::components_large_filter< L >	160
mln::Function_v2b< components_on_border_filter< L > >[external]	
scribo::fun::v2b::components_on_border_filter< L >	162
mln::Function_v2b< components_small_filter< L > >[external]	
scribo::fun::v2b::components_small_filter< L >	164
mln::Function_v2b< label_to_bool< L > >[external]	
scribo::fun::v2b::label_to_bool< L >	165
mln::Function_v2v< highlight< R > >[external]	
scribo::fun::v2v::highlight< R >	166
mln::info	140
mln::Object< E >[external]	
scribo::DMax_Functor< E >	158
scribo::primitive::link::internal::dmax_functor_base< E >	190
scribo::Link_Functor< E >	183
scribo::primitive::link::internal::link_functor_base< L, E >	192
scribo::primitive::link::internal::link_several_dmax_base< L, E >	194
scribo::primitive::link::internal::link_single_dmax_base< L, E >	195
scribo::primitive::link::internal::link_single_dmax_ratio_base< L, F, E >	200
scribo::primitive::link::internal::link_single_dmax_ratio_aligned_base< L, F, E >	197
scribo::primitive::link::internal::link_single_dmax_ratio_aligned_delta_base< L, F, E >	199
scribo::Serializable< E >	202
scribo::SerializeVisitor< E >	203
scribo::doc_serializer< E >	158
mln::Object< component_info< L > >[external]	
scribo::Serializable< component_info< L > >	202
scribo::component_info< L >	149
mln::Object< component_set< L > >[external]	
scribo::Serializable< component_set< L > >	202
scribo::component_set< L >	150
mln::Object< debug_img_visitor< L > >[external]	
scribo::SerializeVisitor< debug_img_visitor< L > >	203
scribo::doc_serializer< debug_img_visitor< L > >	158
scribo::io::img::internal::debug_img_visitor< L >	172
mln::Object< dmax_default >[external]	

scribo::DMax_Functor< dmax_default >	158
scribo::primitive::link::internal::dmax_functor_base< dmax_default >	190
scribo::primitive::link::internal::dmax_default	189
mln::Object< dmax_hrules > [external]	
scribo::DMax_Functor< dmax_hrules >	158
scribo::primitive::link::internal::dmax_functor_base< dmax_hrules >	190
scribo::primitive::link::internal::dmax_hrules	190
mln::Object< dmax_width_and_height > [external]	
scribo::DMax_Functor< dmax_width_and_height >	158
scribo::primitive::link::internal::dmax_functor_base< dmax_width_and_height >	190
scribo::primitive::link::internal::dmax_width_and_height	191
mln::Object< dmax_width_only > [external]	
scribo::DMax_Functor< dmax_width_only >	158
scribo::primitive::link::internal::dmax_functor_base< dmax_width_only >	190
scribo::primitive::link::internal::dmax_width_only	191
mln::Object< document< L > > [external]	
scribo::Serializable< document< L > >	202
scribo::document< L >	159
mln::Object< extended_page_xml_visitor< L > > [external]	
scribo::SerializeVisitor< extended_page_xml_visitor< L > >	203
scribo::doc_serializer< extended_page_xml_visitor< L > >	158
scribo::io::xml::internal::extended_page_xml_visitor< L >	174
mln::Object< full_img_visitor< L > > [external]	
scribo::SerializeVisitor< full_img_visitor< L > >	203
scribo::doc_serializer< full_img_visitor< L > >	158
scribo::io::img::internal::full_img_visitor< L >	172
mln::Object< full_xml_visitor > [external]	
scribo::SerializeVisitor< full_xml_visitor >	203
scribo::doc_serializer< full_xml_visitor >	158
scribo::io::xml::internal::full_xml_visitor	175
mln::Object< line_info< L > > [external]	
scribo::Serializable< line_info< L > >	202
scribo::line_info< L >	176
mln::Object< line_links< L > > [external]	
scribo::Serializable< line_links< L > >	202
scribo::line_links< L >	179
mln::Object< non_text_img_visitor > [external]	
scribo::SerializeVisitor< non_text_img_visitor >	203
scribo::doc_serializer< non_text_img_visitor >	158
scribo::io::img::internal::non_text_img_visitor	173
mln::Object< object_groups< L > > [external]	
scribo::Serializable< object_groups< L > >	202
scribo::object_groups< L >	183
mln::Object< object_links< L > > [external]	
scribo::Serializable< object_links< L > >	202
scribo::object_links< L >	184
mln::Object< page_xml_visitor< L > > [external]	
scribo::SerializeVisitor< page_xml_visitor< L > >	203
scribo::doc_serializer< page_xml_visitor< L > >	158
scribo::io::xml::internal::page_xml_visitor< L >	175
mln::Object< paragraph_set< L > > [external]	
scribo::Serializable< paragraph_set< L > >	202
scribo::paragraph_set< L >	188
mln::Object< several_right_overlap_debug_functor< I, L > > [external]	
scribo::Link_Functor< several_right_overlap_debug_functor< I, L > >	183
scribo::primitive::link::internal::link_functor_base< L, several_right_overlap_debug_functor< I, L > >	192

scribo::primitive::link::internal::link_several_dmax_base< L, several_right_overlap_debug_	
functor< I, L > >	194
mln::Object< single_down_link_debug_functor< I, L > > [external]	
scribo::Link_Functor< single_down_link_debug_functor< I, L > >	183
scribo::primitive::link::internal::link_functor_base< L, single_down_link_debug_functor< I, L > >	192
scribo::primitive::link::internal::link_single_dmax_base< L, single_down_link_debug_functor<	
I, L > >	195
mln::Object< single_left_link_debug_functor< I, L > > [external]	
scribo::Link_Functor< single_left_link_debug_functor< I, L > >	183
scribo::primitive::link::internal::link_functor_base< L, single_left_link_debug_functor< I, L > >	192
scribo::primitive::link::internal::link_single_dmax_base< L, single_left_link_debug_functor< I,	
L > >	195
scribo::primitive::link::internal::link_single_dmax_ratio_base< L, primitive::link::internal::dmax-	
_default, single_left_link_debug_functor< I, L > >	200
mln::Object< single_right_link_debug_functor< I, L > > [external]	
scribo::Link_Functor< single_right_link_debug_functor< I, L > >	183
scribo::primitive::link::internal::link_functor_base< L, single_right_link_debug_functor< I, L > >	192
scribo::primitive::link::internal::link_single_dmax_base< L, single_right_link_debug_functor< I,	
L > >	195
scribo::primitive::link::internal::link_single_dmax_ratio_base< L, primitive::link::internal::dmax-	
_default, single_right_link_debug_functor< I, L > >	200
mln::Object< single_up_link_debug_functor< I, L > > [external]	
scribo::Link_Functor< single_up_link_debug_functor< I, L > >	183
scribo::primitive::link::internal::link_functor_base< L, single_up_link_debug_functor< I, L > >	192
scribo::primitive::link::internal::link_single_dmax_base< L, single_up_link_debug_functor< I, L	
> >	195
mln::Object< text_img_visitor > [external]	
scribo::SerializeVisitor< text_img_visitor >	203
scribo::doc_serializer< text_img_visitor >	158
scribo::io::img::internal::text_img_visitor	173
scribo::binarization::internal::niblack_formula	140
scribo::binarization::internal::niblack_functor< I >	141
scribo::binarization::internal::niblack_functor_fast< I >	141
scribo::binarization::internal::niblack_threshold_functor< I >	142
scribo::binarization::internal::sauvola_formula	143
scribo::binarization::internal::sauvola_functor< I >	143
scribo::binarization::internal::sauvola_ms_functor< I >	144
scribo::binarization::internal::sauvola_threshold_functor< I >	145
scribo::binarization::internal::singh_formula< V >	145
scribo::binarization::internal::singh_functor< I >	146
scribo::binarization::internal::wolf_formula< V >	147
scribo::binarization::internal::wolf_functor< I >	147
scribo::binarization::internal::wolf_functor_fast< I >	148
scribo::component_features_data	149
scribo::debug::arg_data	154
scribo::debug::internal::logger_	155
scribo::debug::opt_data	157
scribo::debug::option_parser	157
scribo::debug::toggle_data	158
scribo::group_info	166
scribo::internal::component_set_data< L >	167
scribo::internal::document_data< L >	167
scribo::internal::line_info_data< L >	168
scribo::internal::line_links_data< L >	169
scribo::internal::line_set_data< L >	169
scribo::internal::object_groups_data< L >	170
scribo::internal::object_links_data< L >	170
scribo::internal::paragraph_set_data< L >	171

scribo::internal::sort_comp_ids< L >	171
scribo::io::xml::internal::color_t	174
scribo::layout::internal::hist_info	176
scribo::layout::internal::node< B >	176
scribo::line_set< L >	180
scribo::paragraph_info< L >	187
scribo::preprocessing::internal::Hough	189
scribo::preprocessing::internal::QCompare	189
scribo::preprocessing::internal::s_angle	189
scribo::toolchain::internal::Toolchain_Functor	208
scribo::toolchain::internal::content_in_doc_functor< I >	203
scribo::toolchain::internal::content_in_hdoc_functor< I >	204
scribo::toolchain::internal::text_in_doc_functor< I >	205
scribo::toolchain::internal::text_in_doc_preprocess_functor< I >	205
scribo::toolchain::internal::text_in_picture_functor< I >	207
scribo::util::integral_sub_sum_sum2_functor< I, S >	208
scribo::util::integral_sum_sum2_functor< V, S >	209
scribo::util::integral_sum_sum2_global_min_functor< V, S >	209
stats< T >	210

Chapter 5

Class Index

5.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

cluster_stats< T >		139
compare_values< T >		139
mln::info		140
scribo::binarization::internal::niblack_formula		140
scribo::binarization::internal::niblack_functor< I >		141
scribo::binarization::internal::niblack_functor_fast< I >		141
scribo::binarization::internal::niblack_threshold_functor< I >		142
scribo::binarization::internal::sauvola_formula		143
scribo::binarization::internal::sauvola_functor< I >		143
scribo::binarization::internal::sauvola_ms_functor< I >		144
scribo::binarization::internal::sauvola_threshold_functor< I >		145
scribo::binarization::internal::singh_formula< V >		145
scribo::binarization::internal::singh_functor< I >		146
scribo::binarization::internal::wolf_formula< V >		147
scribo::binarization::internal::wolf_functor< I >		147
scribo::binarization::internal::wolf_functor_fast< I >		148
scribo::component_features_data		149
scribo::component_info<	L	>
Component information data structure		149
scribo::component_set<	L	>
Represents all the components in a document image		150
scribo::debug::arg_data		154
scribo::debug::internal::logger_		155
scribo::debug::opt_data		157
scribo::debug::option_parser		157
scribo::debug::toggle_data		158
scribo::DMax_Functor<	E	>
Dmax functor concept		158
scribo::doc_serializer<	E	>
Link functor concept		158
scribo::document<	L	>
Represent document data and structure		159
scribo::fun::v2b::components_large_filter<	L	>
Filter Functor		160
scribo::fun::v2b::components_on_border_filter<	L	>
Filter Functor		162
scribo::fun::v2b::components_small_filter<	L	>
Filter Functor		164

scribo::fun::v2b::label_to_bool< L >	165
scribo::fun::v2v::highlight< R >	166
scribo::group_info	166
scribo::internal::component_set_data< L >	167
scribo::internal::document_data< L >	>
Data structure for scribo::document<L>	167
scribo::internal::line_info_data< L >	>
Data structure for scribo::line_info<I>	168
scribo::internal::line_links_data< L >	>
Data structure for scribo::line_links<I>	169
scribo::internal::line_set_data< L >	>
Data structure for scribo::line_set<I>	169
scribo::internal::object_groups_data< L >	>
Data structure for scribo::object_groups<I>	170
scribo::internal::object_links_data< L >	>
Data structure for scribo::object_links<I>	170
scribo::internal::paragraph_set_data< L >	>
Data structure for scribo::paragraph_set<I>	171
scribo::internal::sort_comp_ids< L >	171
scribo::io::img::internal::debug_img_visitor< L >	172
scribo::io::img::internal::full_img_visitor< L >	172
scribo::io::img::internal::non_text_img_visitor	
Save non-text information as an image	173
scribo::io::img::internal::text_img_visitor	173
scribo::io::xml::internal::color_t	174
scribo::io::xml::internal::extended_page_xml_visitor< L >	174
scribo::io::xml::internal::full_xml_visitor	175
scribo::io::xml::internal::page_xml_visitor< L >	>
Save document information as XML	175
scribo::layout::internal::hist_info	176
scribo::layout::internal::node< B >	176
scribo::line_info< L >	176
scribo::line_links< L >	>
Line links representation	179
scribo::line_set< L >	>
Lines container	180
scribo::Link_Functor< E >	>
Link functor concept	183
scribo::object_groups< L >	>
Object group representation	183
scribo::object_links< L >	>
Object links representation	184
scribo::paragraph_info< L >	>
Paragraph structure information	187
scribo::paragraph_set< L >	>
Paragraph container	188
scribo::preprocessing::internal::Hough	189
scribo::preprocessing::internal::QCompare	189
scribo::preprocessing::internal::s_angle	189
scribo::primitive::link::internal::dmax_default	
Base class for dmax functors	189
scribo::primitive::link::internal::dmax_functor_base< E >	>
Base class for dmax functors	190
scribo::primitive::link::internal::dmax_hrules	
Base class for dmax functors	190
scribo::primitive::link::internal::dmax_width_and_height	
Base class for dmax functors	191

scribo::primitive::link::internal::dmax_width_only		
Base class for dmax functors		191
scribo::primitive::link::internal::link_functor_base< L, E >		
Base class for link functors		192
scribo::primitive::link::internal::link_several_dmax_base< L, E >		
Base class for link functors using several anchors and a maximum lookup distance		194
scribo::primitive::link::internal::link_single_dmax_base< L, E >		
Base class for link functors using mass centers and a given max distance		195
scribo::primitive::link::internal::link_single_dmax_ratio_aligned_base< L, F, E >		197
scribo::primitive::link::internal::link_single_dmax_ratio_aligned_delta_base< L, F, E >		199
scribo::primitive::link::internal::link_single_dmax_ratio_base< L, F, E >		
Base class for link functors using bounding box center and a proportional max distance		200
scribo::Serializable< E >		
Concept for objects that can be serialized		202
scribo::SerializeVisitor< E >		
Link functor concept		203
scribo::toolchain::internal::content_in_doc_functor< I >		
Functor analysing and extracting document image content		203
scribo::toolchain::internal::content_in_hdoc_functor< I >		
Functor analysing and extracting content in degraded/historical documents		204
scribo::toolchain::internal::text_in_doc_functor< I >		
Functor extracting text lines from a document image		205
scribo::toolchain::internal::text_in_doc_preprocess_functor< I >		
Functor performing custom preprocessing algorithms on documents		205
scribo::toolchain::internal::text_in_picture_functor< I >		
Localize text in a picture		207
scribo::toolchain::internal::Toolchain_Functor		
Base class for toolchain functors		208
scribo::util::integral_sub_sum_sum2_functor< I, S >		208
scribo::util::integral_sum_sum2_functor< V, S >		209
scribo::util::integral_sum_sum2_global_min_functor< V, S >		209
stats< T >		210

Chapter 6

Module Documentation

6.1 Binarization

Modules

- [Kim](#)
- [Niblack](#)
- [Otsu](#)
- [Sauvola](#)
- [Wolf](#)

6.1.1 Detailed Description

Binarization algorithms.

6.2 Kim

Functions

- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::kim (const Image< I > &input, unsigned window_size, double k)`
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::kim (const Image< I > &input, unsigned window_size)`

6.2.1 Detailed Description

Kim's Binarization implementations. This algorithms performs a first rough binarization on the input (here we use Sauvola's method). Then text lines are roughly detected by grouping connected components. For each text lines, character thickness and x height is computed. Finally, each lines is binarized again, using text features as parameters for adjusting the final threshold.

This algorithms considers that global and local statistics should be used to compute a threshold in text areas.

This is an improvement of Sauvola's method.

This implementation is based on the paper "Multi-Window Binarization of Camera Image for Document Recognition", In-Jung Kim, Proceedings of the 9th International Workshop on Frontiers in Handwriting Recognition (IWFHR-9 2004)

6.2.2 Function Documentation

6.2.2.1 `template<typename I > mln::trait::ch_value< I, bool >::ret scribo::binarization::kim (const Image< I > &input, unsigned window_size, double k)`

Kim's binarization.

This algorithms performs a first rough binarization on the input (here we use Sauvola's method). Then text lines are roughly detected by grouping connected components. For each text lines, character thickness and x height is computed. Finally, each lines is binarized again, using text features as parameters for adjusting the final threshold.

This algorithms considers that global and local statistics should be used to compute a threshold in text areas.

This is an improvement of Sauvola's method.

This implementation is based on the paper "Multi-Window Binarization of Camera Image for Document Recognition", In-Jung Kim, Proceedings of the 9th International Workshop on Frontiers in Handwriting Recognition (IWFHR-9 2004)

Parameters

<code>in</code>	<code>input</code>	A gray-level image.
<code>in</code>	<code>window_size</code>	the window size to be used for the first binarization.
<code>in</code>	<code>k</code>	Sauvola's formula parameter.

6.2.2.2 `template<typename I > mln::trait::ch_value< I, bool >::ret scribo::binarization::kim (const Image< I > &input, unsigned window_size)`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. k is set to `SCRIBO_DEFAULT_SAUVOLA_K`.

6.3 Niblack

Functions

- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::niblack` (const `Image< I >` &input, `unsigned` window_size, `double` K)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::niblack` (const `Image< I >` &input, `unsigned` window_size)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::niblack` (const `Image< I >` &input)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::niblack_fast` (const `Image< I >` &input, `unsigned` window_size, `double` K)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::niblack_fast` (const `Image< I >` &input, `unsigned` window_size)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::niblack_fast` (const `Image< I >` &input)
- `template<typename I, typename J >`
`mln::trait::ch_value< I,`
`value::int_u8 >::ret scribo::binarization::niblack_threshold` (const `Image< I >` &input, `unsigned` window_ -
size, `double` K)
- `template<typename I >`
`mln::trait::ch_value< I,`
`value::int_u8 >::ret scribo::binarization::niblack_threshold` (const `Image< I >` &input, `unsigned` window_ -
size)
- `template<typename I >`
`mln::trait::ch_value< I,`
`value::int_u8 >::ret scribo::binarization::niblack_threshold` (const `Image< I >` &input)

6.3.1 Detailed Description

Niblack's Binarization implementations.

6.3.2 Function Documentation

6.3.2.1 `template<typename I > mln::trait::ch_value< I, bool >::ret scribo::binarization::niblack (const Image< I > &input, unsigned window_size, double K)`

Convert an image into a binary image.

Parameters

<code>in</code>	<code>input</code>	An image.
<code>in</code>	<code>window_size</code>	The window size.
<code>in</code>	<code>K</code>	Niblack's formulae constant.

Returns

A binary image.

6.3.2.2 `template<typename I> mln::trait::ch_value< I, bool >::ret scribo::binarization::niblack (const Image< I > & input, unsigned window_size)`

Convert an image into a binary image.

Sauvola's formulae constant K is set to SCRIBO_DEFAULT_NIBLACK_K.

Parameters

<code>in</code>	<code>input</code>	An image.
<code>in</code>	<code>window_size</code>	The window size.

Returns

A binary image.

6.3.2.3 `template<typename I> mln::trait::ch_value< I, bool >::ret scribo::binarization::niblack (const Image< I > & input)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. The window size is set to 11.

6.3.2.4 `template<typename I> mln::trait::ch_value< I, bool >::ret scribo::binarization::niblack_fast (const Image< I > & input, unsigned window_size, double K)`

Convert an image into a binary image.

This implementation gives an approximation of the results. **It** is faster than the original implementation thanks to the use of integral images.

Parameters

<code>in</code>	<code>input</code>	An image.
<code>in</code>	<code>window_size</code>	The window size.
<code>in</code>	<code>K</code>	Sauvola's formulae constant.

Returns

A binary image.

6.3.2.5 `template<typename I> mln::trait::ch_value< I, bool >::ret scribo::binarization::niblack_fast (const Image< I > & input, unsigned window_size)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. Sauvola's formulae constant K is set to SCRIBO_DEFAULT_NIBLACK_K.

6.3.2.6 `template<typename I> mln::trait::ch_value< I, bool >::ret scribo::binarization::niblack_fast (const Image< I > & input)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. The window size is set to 11.

```
6.3.2.7 template<typename I , typename J > mIn::trait::ch_value< I , value::int_u8 >::ret
scribo::binarization::niblack_threshold ( const Image< I > & input, unsigned window_size, double K )
```

Compute an image of local threshold using Niblack algorithm.

Parameters

in	<i>input</i>	A gray level image.
in	<i>window_size</i>	The window size.
in	<i>K</i>	Controls the threshold value in the local window.

Returns

An image of local thresholds.

```
6.3.2.8 template<typename I > mIn::trait::ch_value< I , value::int_u8 >::ret scribo::binarization::niblack_threshold (
const Image< I > & input, unsigned window_size )
```

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. *K* is set to 0.34.

```
6.3.2.9 template<typename I > mIn::trait::ch_value< I , value::int_u8 >::ret scribo::binarization::niblack_threshold (
const Image< I > & input )
```

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. The window size is set to 11.

6.4 Otsu

Functions

- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::otsu (const Image< I > &input)`
- `template<typename I >`
`I::value scribo::binarization::otsu_threshold (const Image< I > &input)`

6.4.1 Detailed Description

Otsu's Binarization implementations. This implementation is based on Ocropus's implementation. <http://code.google.com/p/ocropus/>

6.4.2 Function Documentation

6.4.2.1 `template<typename I > mln::trait::ch_value< I, bool >::ret scribo::binarization::otsu (const Image< I > &input)`

An implementation of Otsu's binarization algorithm.

This implementation is based on Ocropus's implementation. <http://code.google.com/p/ocropus/>

Parameters

<code>in</code>	<code>input</code>	A gray-scale image.
-----------------	--------------------	---------------------

Returns

A binary image. True for foreground, False for background.

6.4.2.2 `template<typename I > I::value scribo::binarization::otsu_threshold (const Image< I > &input)`

Compute a global binarization threshold using Otsu's algorithm.

This implementation is based on Ocropus's implementation. <http://code.google.com/p/ocropus/>

Parameters

<code>in</code>	<code>input</code>	A gray-scale image.
-----------------	--------------------	---------------------

Returns

A global threshold value.

6.5 Sauvola

Functions

- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::sauvola` (const `Image< I >` &input, `unsigned` window_size, `double` K)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::sauvola` (const `Image< I >` &input, `unsigned` window_size)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::sauvola` (const `Image< I >` &input)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::sauvola_ms` (const `Image< I >` &input_1, `unsigned` w_1, `unsigned` s, `image2d<`
`mln::util::couple< double, double >` > &integral_sum_sum_2)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::sauvola_ms` (const `Image< I >` &input_1, `unsigned` w_1, `unsigned` s)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::sauvola_ms` (const `Image< I >` &input_1, `unsigned` w_1)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::sauvola_ms_split` (const `Image< I >` &input_1, `unsigned` w_1, `unsigned` s, `unsigned`
`min_ntrue`, `double` k2, `double` k3, `double` k4)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::sauvola_ms_split` (const `Image< I >` &input_1, `unsigned` w_1, `unsigned` s, `unsigned`
`min_ntrue`, `double` K)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::sauvola_ms_split` (const `Image< I >` &input_1, `unsigned` w_1, `unsigned` s, `unsigned`
`min_ntrue`)
- `template<typename I, typename J >`
`mln::trait::ch_value< I,`
`value::int_u8 >::ret scribo::binarization::sauvola_threshold` (const `Image< I >` &input, `unsigned` window_
`size`, `double` K)
- `template<typename I >`
`mln::trait::ch_value< I,`
`value::int_u8 >::ret scribo::binarization::sauvola_threshold` (const `Image< I >` &input, `unsigned` window_
`size`)
- `template<typename I >`
`mln::trait::ch_value< I,`
`value::int_u8 >::ret scribo::binarization::sauvola_threshold` (const `Image< I >` &input)

6.5.1 Detailed Description

Sauvola's Binarization implementations. It is well suited for classical document image binarization.

Scribo provides three variants of Sauvola's binarization algorithm :

- `scribo::binarization::sauvola` , the standard implementation.
- `scribo::binarization::sauvola_ms` , a multi-scale implementation.

- [scribo::binarization::sauvola_ms_split](#) , a multi-scale implementation running on each channel of a RGB image and merging the results into a single binary image.

These algorithms are based on the following article: [5] . All of them are implemented using integral images for best performance [6].

Sauvola's algorithm is a local thresholding method. Based on the mean and the standard deviation of a sliding centered window, it computes a local threshold for each pixel.

The following formula is used:

$$T(p) = m(p) + [1 + k * (\frac{s(p)}{R} - 1)]$$

Where p is the central window point, $m(p)$ and $s(p)$ are, respectively, the mean and the standard deviation of the window centered to point p .

k is user-defined parameter. According to [1], this parameter is set to 0.34 . However, it is possible to pass a different value and change the object sharpness.

The window size is also a user-defined parameter. In [scribo::binarization::sauvola](#), it has to be adapted to the size of the binarized objects. In case of a document image with mixed objects of various size, we advice you to use the multi-scale version [scribo::binarization::sauvola_ms](#). In this version, the window size parameter is adapted at each scale and both large and small objects are correctly retrieved at the same time.

--	--	--	--

[scribo::binarization::sauvola_threshold](#) variants return an image of thresholds. The latter can be used afterwards to binarize the input image.

See Also

[src/binarization/sauvola.cc](#) [src/binarization/sauvola_ms.cc](#) [sauvola_ms_split.cc](#)

6.5.2 comparison

Compare results between all Sauvola's implementations. The window size is set to 51 and k is set to 0.34 .

6.5.3 Function Documentation

6.5.3.1 `template<typename I> mIn::trait::ch_value< I, bool >::ret scribo::binarization::sauvola (const Image< I > & input, unsigned window_size, double K)`

Convert an image into a binary image.

Parameters

in	<i>input</i>	A greyscale image.
in	<i>window_size</i>	The window size.
in	<i>K</i>	Sauvola's formulae constant.

Returns

A binary image.

6.5.3.2 `template<typename I> mIn::trait::ch_value< I, bool >::ret scribo::binarization::sauvola (const Image< I > & input, unsigned window_size)`

Convert an image into a binary image.

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. Sauvola's formulae constant K is set to 0.34.

Parameters

in	<i>input</i>	A grayscale image.
in	<i>window_size</i>	The window size.

Returns

A binary image.

6.5.3.3 `template<typename I> mIn::trait::ch_value< I, bool >::ret scribo::binarization::sauvola (const Image< I > & input)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. The window size is set to 11.

6.5.3.4 `template<typename I> mIn::trait::ch_value< I, bool >::ret scribo::binarization::sauvola_ms (const Image< I > & input_1, unsigned w_1, unsigned s, image2d< mIn::util::couple< double, double > > & integral_sum_sum_2)`

Binarize an image using a multi-scale implementation of Sauvola's algorithm.

Parameters

in	<i>input_1</i>	A grayscale image.
in	<i>w_1</i>	The window size used to compute stats.
in	<i>s</i>	The scale factor used for the first subscaling (usually 2 or 3 is enough).
out	<i>integral_sum - sum_2</i>	Integral image of sum and squared sum.

Sauvola's formula parameter K is set to 0.34. *w_1* and *lambda_min_1* are expressed according to the image at scale 0, i.e. the original size.

Returns

A Boolean image.

In this implementation, the input image is subscaled at three different scales. For each subscale image, Sauvola's is applied and relevant objects are preserved. The results of the three scales is merged and areas of the input image are marked with the appropriate scale to use for final binarization. Finally, the input image is binarized, using, for each pixel, the correct window size of the scale where it has been retrieved.

This implementation is very useful for document image with heterogeneous object sizes.

Its implementation is meant to be the fastest as possible. For instance, statistics computation is based on integral images [6].

6.5.3.5 `template<typename I> mln::trait::ch_value< I , bool >::ret scribo::binarization::sauvola_ms (const Image< I > & input_1, unsigned w_1, unsigned s)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. The integral image is not returned.

K is set to 0.34.

6.5.3.6 `template<typename I> mln::trait::ch_value< I , bool >::ret scribo::binarization::sauvola_ms (const Image< I > & input_1, unsigned w_1)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. The integral image is not returned.

K is set to 0.34. s is set to 3.

6.5.3.7 `template<typename I> mln::trait::ch_value< I , bool >::ret scribo::binarization::sauvola_ms_split (const Image< I > & input_1, unsigned w_1, unsigned s, unsigned min_ntrue, double k2, double k3, double k4)`

Binarize a color image merging the binarization of each component using Sauvola's algorithm.

Parameters

in	<i>input_1</i>	A color image.
in	<i>w_1</i>	The window size used to compute stats.
in	<i>s</i>	The scale factor used for the first subscaling.
in	<i>min_ntrue</i>	A site is set to 'True' in the output if it is set to 'True' at least <i>min_ntrue</i> components. Possible values: 1, 2, 3.
in	<i>k2</i>	Sauvola's formula parameter.
in	<i>k3</i>	Sauvola's formula parameter.
in	<i>k4</i>	Sauvola's formula parameter.

w_1 is expressed according to the image at scale 0, i.e. the original size.

Returns

A Boolean image.

6.5.3.8 `template<typename I> mln::trait::ch_value< I , bool >::ret scribo::binarization::sauvola_ms_split (const Image< I > & input_1, unsigned w_1, unsigned s, unsigned min_ntrue, double K)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. *k2*, *k3* and *k4* are set to *K*.

6.5.3.9 `template<typename I> mln::trait::ch_value< I , bool >::ret scribo::binarization::sauvola_ms_split (const Image< I > & input_1, unsigned w_1, unsigned s, unsigned min_ntrue)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. *k2*, *k3* and *k4* are set to 0.34.

6.5.3.10 `template<typename I , typename J> mln::trait::ch_value< I , value::int_u8 >::ret scribo::binarization::sauvola_threshold (const Image< I > & input, unsigned window_size, double K)`

Compute an image of local threshold using Sauvola algorithm.

Parameters

in	<i>input</i>	A gray level image.
in	<i>window_size</i>	The window size.
in	<i>K</i>	Controls the threshold value in the local window.

Returns

An image of local thresholds.

6.5.3.11 `template<typename I> mIn::trait::ch_value< I , value::int_u8 >::ret scribo::binarization::sauvola_threshold (const Image< I > & input, unsigned window_size)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. *K* is set to 0.34.

6.5.3.12 `template<typename I> mIn::trait::ch_value< I , value::int_u8 >::ret scribo::binarization::sauvola_threshold (const Image< I > & input)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. The window size is set to 11.

6.6 Wolf

Functions

- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::wolf` (const `Image< I >` &input, `unsigned` window_size, `double` K)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::wolf` (const `Image< I >` &input, `unsigned` window_size)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::wolf` (const `Image< I >` &input)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::wolf_fast` (const `Image< I >` &input, `unsigned` window_size, `double` K)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::wolf_fast` (const `Image< I >` &input, `unsigned` window_size)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::binarization::wolf_fast` (const `Image< I >` &input)

6.6.1 Detailed Description

Wolf's Binarization implementations.

6.6.2 Function Documentation

6.6.2.1 `template<typename I > mln::trait::ch_value< I, bool >::ret scribo::binarization::wolf (const Image< I > &input, unsigned window_size, double K)`

Convert an image into a binary image.

Parameters

<code>in</code>	<code>input</code>	An image.
<code>in</code>	<code>window_size</code>	The window size.
<code>in</code>	<code>K</code>	Wolf's formulae constant.

Returns

A binary image.

This implementation is based on article "Text Localization, Enhancement and Binarization in Multimedia Documents", Christian Wolf, Jean-Michel Jolion, Françoise Chassaing, ICPR 2002.

6.6.2.2 `template<typename I > mln::trait::ch_value< I, bool >::ret scribo::binarization::wolf (const Image< I > &input, unsigned window_size)`

Convert an image into a binary image.

Wolf's formulae constant K is set to 0.34.

Parameters

<code>in</code>	<code>input</code>	An image.
<code>in</code>	<code>window_size</code>	The window size.

Returns

A binary image.

6.6.2.3 `template<typename I> mln::trait::ch_value< I, bool >::ret scribo::binarization::wolf (const Image< I > & input)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. The window size is set to 11.

6.6.2.4 `template<typename I> mln::trait::ch_value< I, bool >::ret scribo::binarization::wolf_fast (const Image< I > & input, unsigned window_size, double K)`

Convert an image into a binary image.

Parameters

<code>in</code>	<code>input</code>	An image.
<code>in</code>	<code>window_size</code>	The window size.
<code>in</code>	<code>K</code>	Wolf's formulae constant.

Returns

A binary image.

This implementation is based on article "Text Localization, Enhancement and Binarization in Multimedia Documents", Christian Wolf, Jean-Michel Jolion, Françoise Chassaing, ICPR 2002.

This implementation gives an approximation of the results. **It** is faster than the original implementation thanks to the use of integral images.

6.6.2.5 `template<typename I> mln::trait::ch_value< I, bool >::ret scribo::binarization::wolf_fast (const Image< I > & input, unsigned window_size)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. Wolf's formulae constant K is set to 0.34.

6.6.2.6 `template<typename I> mln::trait::ch_value< I, bool >::ret scribo::binarization::wolf_fast (const Image< I > & input)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. The window size is set to 11.

6.7 Data structures

Classes

- class `scribo::component_set< L >`
Represents all the components in a document image.
- class `scribo::document< L >`
Represent document data and structure.
- class `scribo::line_links< L >`
Line links representation.
- class `scribo::line_set< L >`
Lines container.
- class `scribo::object_groups< L >`
Object group representation.
- class `scribo::object_links< L >`
Object links representation.
- class `scribo::paragraph_set< L >`
Paragraph container.

6.7.1 Detailed Description

Scribo high level data structures. For the moment, Scribo provides bottom-up approaches to extract data from a document. From connected components, thanks to different algorithms, lines and paragraphs are reconstructed. It implies the following workflow (see figure below).

At each level, it is usually possible to "filter" the current results using routines in namespace `scribo::filter`. False positive can then be invalidated.

Data structures at a specific step, usually keep track of previous parent structures. For instance, `object_links` keeps a reference to the `component_set` it is based on.

These structures are iterable and usually store several attributes for their corresponding elements.

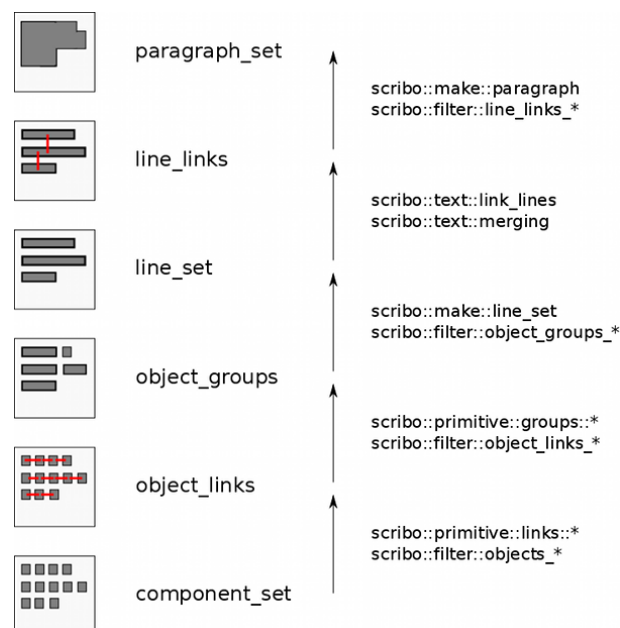


Figure 6.1: Data structure construction workflow and useful routines/namespaces for text extraction.

6.8 Debug

Functions

- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::alignment_decision_image (const **Image**< I > &input, const object_links< L > &links, const object_links< L > &filtered_links, const anchor::Type &anchor)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::bboxes_enlarged_image (const **Image**< I > &input, const line_set< L > &lines, const value::rgb8 &text_value, const value::rgb8 &non_text_value)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::bboxes_enlarged_image (const **Image**< I > &input, const line_set< L > &lines)
- `template<typename I>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::bboxes_image (const **Image**< I > &input, const mln::util::array< box< typename I::site > > &bboxes, const value::rgb8 &value)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::bboxes_image (const **Image**< I > &input, const line_set< L > &lines, const value::rgb8 &value)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::bboxes_image (const **Image**< I > &input, const line_set< L > &lines)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::bboxes_image (const **Image**< I > &input, const component_set< L > &comps, const value::rgb8 &value)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::char_space_image (const **Image**< I > &input, const line_set< L > &line, const value::rgb8 &v=literal::cyan)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::decision_image (const **Image**< I > &input, const object_groups< L > &groups, const object_groups< L > &filtered_groups, anchor::Type anchor)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::decision_image (const **Image**< I > &input, const object_links< L > &links, const object_links< L > &filtered_links, anchor::Type anchor)
- `template<typename I>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::highlight_text_area (const **Image**< I > &input, const mln::util::array< box< typename I::site > > &bbox)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::highlight_text_area (const **Image**< I > &input, const line_set< L > &lines)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::highlight_text_area (const **Image**< I > &input, const scribo::component_set< L > &components)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::line_info_image (const **Image**< I > &input, const line_set< L > &line)

- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::linked_bboxes_image (const **Image**< I > &input, const object_links< L > &array, const value::rgb8 &box_value, const value::rgb8 &link_value, anchor::Type anchor)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::linked_bboxes_image (const **Image**< I > &input, const object_links< L > &array, const value::rgb8 &box_value, const value::rgb8 &link_value)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::linked_bboxes_image (const **Image**< I > &input, const object_links< L > &left_link, const object_links< L > &right_link, const value::rgb8 &box_value, const value::rgb8 &link_value, anchor::Type anchor)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::linked_bboxes_image (const **Image**< I > &input, const object_links< L > &left_link, const object_links< L > &right_link, const value::rgb8 &box_value, const value::rgb8 &left_link_value, const value::rgb8 &right_link_value, const value::rgb8 &validated_link_value, anchor::Type anchor)
- `template<typename I, typename L, typename G>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::linked_bboxes_image (const **Image**< I > &input, const **Graph**< G > &g, const value::rgb8 &box_value, const value::rgb8 &link_value, anchor::Type anchor)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::links_decision_image (const **Image**< I > &input_, const object_links< L > &links, const object_links< L > &filtered_links)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::links_image (const **Image**< I > &input_, const object_links< L > &links, anchor::Type anchor, **bool** draw_bboxes=true)
- `scribo::debug::internal::logger_ & scribo::debug::logger ()`
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::looks_like_a_text_line_image (const **Image**< I > &input, const line_set< L > &lines, const value::rgb8 &text_value, const value::rgb8 &non_text_value)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::looks_like_a_text_line_image (const **Image**< I > &input, const line_set< L > &lines)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::mean_and_base_lines_image (const **Image**< I > &input, const line_set< L > &lines, const value::rgb8 &bbox_value, const value::rgb8 &meanline_value, const value::rgb8 &baseline_value)
- `template<typename I, typename L>`
mln::trait::ch_value< I,
value::rgb8 >::ret scribo::debug::mean_and_base_lines_image (const **Image**< I > &input, const line_set< L > &lines)
- `template<typename L, typename L2>`
void scribo::debug::save_comp_diff (const component_set< L > &comps_ref, const component_set< L2 > &comps_new, const std::string &filename)
- `template<typename I>`
void scribo::debug::save_label_image (const **Image**< I > &lbl, const typename I::value &nlabels, const char *filename)
- `template<typename I>`
void scribo::debug::save_table_image (const **Image**< I > &input_, mln::util::couple< mln::util::array< **box**< typename I::site > >, mln::util::array< **box**< typename I::site > > > tableboxes, const value::rgb8 &bbox_color, const std::string &filename)

- `template<typename S >`
`void scribo::debug::save_table_image (const Site_Set< S > &input_domain, mln::util::couple< mln::util::array< box< typename S::site > >, mln::util::array< box< typename S::site > > > tableboxes, const value::rgb8 &bg_color, const value::rgb8 &bbox_color, const std::string &filename)`
- `template<typename I, typename L >`
`mln::trait::concrete< I >::ret scribo::debug::text_areas_image (const Image< I > &input_rgb, const scribo::component_set< L > &comps)`
- `template<typename L >`
`image2d< value::rgb8 > scribo::debug::text_color_image (const document< L > &doc)`

6.8.1 Detailed Description

Debug routines.

6.8.2 Function Documentation

6.8.2.1 `template<typename I, typename L > mln::trait::ch_value< I, value::rgb8 >::ret scribo::debug::alignment_decision_image (const Image< I > &input, const object_links< L > &links, const object_links< L > &filtered_links, const anchor::Type &anchor)`

Save a color image showing the difference between to object links.

Parameters

<code>in</code>	<code>input</code>	An image. It's value type must be convertible towards <code>rgb8</code> .
<code>in</code>	<code>links</code>	Object links information.
<code>in</code>	<code>filtered_links</code>	A copy of <code>links</code> which have been filtered.
<code>in</code>	<code>anchor</code>	Anchor from where the links are drawn.

Returns

A color image. Non filtered links are drawn in green. Others are drawn in red.

6.8.2.2 `template<typename I, typename L > mln::trait::ch_value< I, value::rgb8 >::ret scribo::debug::bboxes_enlarged_image (const Image< I > &input, const line_set< L > &lines, const value::rgb8 &text_value, const value::rgb8 &non_text_value)`

Compute an image of enlarged component bounding boxes.

This check whether each line looks like a text line. If it is a text line, its extended bounding box is drawn, otherwise, it is normal bounding box.

This routine uses `scribo::internal::looks_like_a_text_line` to check if a component looks like a text line.

Parameters

<code>in</code>	<code>input</code>	An image convertible towards a color image.
<code>in</code>	<code>lines</code>	A line set.
<code>in</code>	<code>text_value</code>	The color used to draw bounding boxes of components looking like a text line.
<code>in</code>	<code>non_text_value</code>	The color used to draw bounding boxes of components NOT looking like a text line.

Returns

A color image.

6.8.2.3 `template<typename I , typename L > mln::trait::ch_value< I , value::rgb8 >::ret scribo::debug::bboxes_enlarged_image (const Image< I > & input, const line_set< L > & lines)`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `text_value` is set to `literal::green`.

`non_text_value` is set to `literal::red`.

6.8.2.4 `template<typename I > mln::trait::ch_value< I , value::rgb8 >::ret scribo::debug::bboxes_image (const Image< I > & input, const mln::util::array< box< typename I::site > > & bboxes, const value::rgb8 & value)`

Draw a list of bounding boxes.

6.8.2.5 `template<typename I , typename L > mln::trait::ch_value< I , value::rgb8 >::ret scribo::debug::bboxes_image (const Image< I > & input, const line_set< L > & lines, const value::rgb8 & value)`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

6.8.2.6 `template<typename I , typename L > mln::trait::ch_value< I , value::rgb8 >::ret scribo::debug::bboxes_image (const Image< I > & input, const line_set< L > & lines) [inline]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `value` is set to `literal::red`.

6.8.2.7 `template<typename I , typename L > mln::trait::ch_value< I , value::rgb8 >::ret scribo::debug::bboxes_image (const Image< I > & input, const component_set< L > & comps, const value::rgb8 & value) [inline]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

6.8.2.8 `template<typename I , typename L > mln::trait::ch_value< I , value::rgb8 >::ret scribo::debug::char_space_image (const Image< I > & input, const line_set< L > & line, const value::rgb8 & v = literal::cyan)`

Draw inter character space.

6.8.2.9 `template<typename I , typename L > mln::trait::ch_value< I , value::rgb8 >::ret scribo::debug::decision_image (const Image< I > & input, const object_groups< L > & groups, const object_groups< L > & filtered_groups, anchor::Type anchor)`

Save a color image showing the difference between to object groups.

Parameters

<code>in</code>	<code>input</code>	An image. It's value type must be convertible towards <code>rgb8</code> .
<code>in</code>	<code>groups</code>	Object groups information.
<code>in</code>	<code>filtered_groups</code>	A copy of <code>groups</code> which have been filtered.
<code>in</code>	<code>anchor</code>	Anchor from where the links are drawn.

Returns

A color image. Components part of a validated group are drawn in green with their bounding box. Otherwise, they are drawn in red.

6.8.2.10 `template<typename I, typename L> mln::trait::ch_value< I, value::rgb8 >::ret scribo::debug::decision_image (const Image< I > & input, const object_links< L > & links, const object_links< L > & filtered_links, anchor::Type anchor)`

Save a color image showing the difference between to object links.

Parameters

in	<i>input</i>	An image. It's value type must be convertible towards rgb8.
in	<i>links</i>	Object links information.
in	<i>filtered_links</i>	A copy of <code>links</code> which have been filtered.
in	<i>anchor</i>	Anchor from where the links are drawn.

Returns

A color image. Non filtered links are drawn in green. Others are drawn in red.

6.8.2.11 `template<typename I> mln::trait::ch_value< I, value::rgb8 >::ret scribo::debug::highlight_text_area (const Image< I > & input, const mln::util::array< box< typename I::site > > & bbox)`

Darken an image and highlight valid lines.

Parameters

in	<i>input</i>	An image.
in	<i>bbox</i>	An array of text area bounding boxes.

Returns

a color image with highlighted text areas.

6.8.2.12 `template<typename I, typename L> mln::trait::ch_value< I, value::rgb8 >::ret scribo::debug::highlight_text_area (const Image< I > & input, const line_set< L > & lines)`

Darken an image and highlight valid lines.

Parameters

in	<i>input</i>	An image.
in	<i>lines</i>	A line set.

Returns

a color image with highlighted text areas.

6.8.2.13 `template<typename I, typename L> mln::trait::ch_value< I, value::rgb8 >::ret scribo::debug::highlight_text_area (const Image< I > & input, const scribo::component_set< L > & components)`

Darken an image and highlight valid lines.

This overload is useful if you do know that all the components are text.

Parameters

in	<i>input</i>	An image.
in	<i>components</i>	A component set.

Returns

a color image with highlighted text areas.

6.8.2.14 `template<typename I, typename L> mIn::trait::ch_value< I, value::rgb8 >::ret scribo::debug::line_info_image (const Image< I > & input, const line_set< L > & line)`

Draw typographic information from lines.

6.8.2.15 `template<typename I, typename L> mIn::trait::ch_value< I, value::rgb8 >::ret scribo::debug::linked_bboxes_image (const Image< I > & input, const object_links< L > & array, const value::rgb8 & box_value, const value::rgb8 & link_value, anchor::Type anchor)`

Compute the line of components links image.

Parameters

in, out	<i>input</i>	The binary from where the components are extracted.
in	<i>array</i>	Components links.
in	<i>box_value</i>	Value used to draw line bounding boxes.
in	<i>link_value</i>	Value used to draw line links.
in	<i>anchor</i>	Anchor from where the links are drawn.

6.8.2.16 `template<typename I, typename L> mIn::trait::ch_value< I, value::rgb8 >::ret scribo::debug::linked_bboxes_image (const Image< I > & input, const object_links< L > & array, const value::rgb8 & box_value, const value::rgb8 & link_value)`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. The default anchor type is set to `anchor::Center`.

6.8.2.17 `template<typename I, typename L> mIn::trait::ch_value< I, value::rgb8 >::ret scribo::debug::linked_bboxes_image (const Image< I > & input, const object_links< L > & left_link, const object_links< L > & right_link, const value::rgb8 & box_value, const value::rgb8 & link_value, anchor::Type anchor)`

Compute the line of components left and right links image.

Parameters

in, out	<i>input</i>	The binary from where the components are extracted.
in	<i>left_link</i>	Components left links.
in	<i>right_link</i>	Components right links.
in	<i>box_value</i>	Value used to draw line bounding boxes.
in	<i>link_value</i>	Value used to draw line links.
in	<i>anchor</i>	Anchor from where the links are drawn.

6.8.2.18 `template<typename I, typename L> mIn::trait::ch_value< I, value::rgb8 >::ret scribo::debug::linked_bboxes_image (const Image< I> & input, const object_links< L> & left_link, const object_links< L> & right_link, const value::rgb8 & box_value, const value::rgb8 & left_link_value, const value::rgb8 & right_link_value, const value::rgb8 & validated_link_value, anchor::Type anchor) [inline]`

Compute the line of components left and right links image.

Draw also validated links.

Parameters

in, out	<i>input</i>	The binary from where the components are extracted.
in	<i>left_link</i>	Components left links.
in	<i>right_link</i>	Components right links.
in	<i>box_value</i>	Value used to draw line bounding boxes.
in	<i>left_link_value</i>	Value used to draw line left links.
in	<i>right_link_value</i>	Value used to draw line left links.
in	<i>validated_link_value</i>	Value used to draw line validated links.
in	<i>anchor</i>	Anchor from where the links are drawn.

6.8.2.19 `template<typename I, typename L, typename G> mIn::trait::ch_value< I, value::rgb8 >::ret scribo::debug::linked_bboxes_image (const Image< I> & input, const Graph< G> & g, const value::rgb8 & box_value, const value::rgb8 & link_value, anchor::Type anchor)`

Compute the line link graph image.

Parameters

in, out	<i>input</i>	The binary from where the components are extracted.
in	<i>g</i>	The link graph.
in	<i>box_value</i>	Value used to draw line bounding boxes.
in	<i>link_value</i>	Value used to draw line links.
in	<i>anchor</i>	Anchor from where the links are drawn.

6.8.2.20 `template<typename I, typename L> mIn::trait::ch_value< I, value::rgb8 >::ret scribo::debug::links_decision_image (const Image< I> & input, const object_links< L> & links, const object_links< L> & filtered_links)`

Save a color image showing the difference between to object groups.

6.8.2.21 `template<typename I, typename L> mIn::trait::ch_value< I, value::rgb8 >::ret scribo::debug::links_image (const Image< I> & input, const object_links< L> & links, anchor::Type anchor, bool draw_bboxes = true)`

Save a color image showing components links.

6.8.2.22 `scribo::debug::internal::logger_ & scribo::debug::logger ()`

Return a reference to the logger.

6.8.2.23 `template<typename I, typename L> mIn::trait::ch_value< I, value::rgb8 >::ret scribo::debug::looks_like_a_text_line_image (const Image< I > & input, const line_set< L > & lines, const value::rgb8 & text_value, const value::rgb8 & non_text_value)`

Compute an image where components are drawn differently whether they look like a line or not.

Parameters

in	<i>input</i>	An image convertible towards a color image.
in	<i>lines</i>	A line set.
in	<i>text_value</i>	The color used to draw bounding boxes of components looking like a text line.
in	<i>non_text_value</i>	The color used to draw bounding boxes of components NOT looking like a text line.

Returns

A color image.

6.8.2.24 `template<typename I, typename L> mIn::trait::ch_value< I, value::rgb8 >::ret scribo::debug::looks_like_a_text_line_image (const Image< I > & input, const line_set< L > & lines)`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. *text_value* is set to `literal::green`.

non_text_value is set to `literal::red`.

6.8.2.25 `template<typename I, typename L> mIn::trait::ch_value< I, value::rgb8 >::ret scribo::debug::mean_and_base_lines_image (const Image< I > & input, const line_set< L > & lines, const value::rgb8 & bbox_value, const value::rgb8 & meanline_value, const value::rgb8 & baseline_value)`

Compute a color image showing the mean and the base lines of the text lines.

The mean line is drawn with a dashed line. The base line is drawn with a plain line.

Parameters

in	<i>input</i>	An image convertible towards a color image.
in	<i>lines</i>	A line set.
in	<i>bbox_value</i>	Value used to draw lines bounding boxes.
in	<i>meanline_value</i>	Value used to draw mean lines.
in	<i>baseline_value</i>	Value used to draw base lines.

Returns

A color image.

6.8.2.26 `template<typename I, typename L> mIn::trait::ch_value< I, value::rgb8 >::ret scribo::debug::mean_and_base_lines_image (const Image< I > & input, const line_set< L > & lines)`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. *text_value* is set to `literal::green`.

non_text_value is set to `literal::red`.

6.8.2.27 `template<typename L , typename L2 > void scribo::debug::save_comp_diff (const component_set< L > & comps_ref, const component_set< L2 > & comps_new, const std::string & filename)`

Show the difference between two object images.

Parameters

in	<i>comps_ref</i>	A component set.
in	<i>comps_new</i>	Another component set.
in	<i>filename</i>	The output filename.

6.8.2.28 `template<typename I > void scribo::debug::save_label_image (const Image< I > & lbl, const typename I::value & nlabels, const char * filename)`

Save a labeled image in a color image.

Parameters

in	<i>lbl</i>	A label image.
in	<i>nlabels</i>	The number of labels.
in	<i>filename</i>	The output file name.

6.8.2.29 `template<typename I > void scribo::debug::save_table_image (const Image< I > & input_, mIn::util::couple< mIn::util::array< box< typename I::site > >, mIn::util::array< box< typename I::site > > > > tableboxes, const value::rgb8 & bbox_color, const std::string & filename)`

Save lines bounding boxes in a copy of `input_`.

Bounding boxes are displayed with `bbox_color`.

6.8.2.30 `template<typename S > void scribo::debug::save_table_image (const Site_Set< S > & input_domain, mIn::util::couple< mIn::util::array< box< typename S::site > >, mIn::util::array< box< typename S::site > > > > > tableboxes, const value::rgb8 & bg_color, const value::rgb8 & bbox_color, const std::string & filename)`

Save lines bounding boxes in an image defined on `input_domain` filled with `bg_color`.

Bounding boxes are displayed with `bbox_color`.

6.8.2.31 `template<typename I , typename L > mIn::trait::concrete< I >::ret scribo::debug::text_areas_image (const Image< I > & input_rgb, const scribo::component_set< L > & comps)`

Compute an image including detected text areas only.

6.8.2.32 `template<typename L > image2d<value::rgb8> scribo::debug::text_color_image (const document< L > & doc)`

Draw text components with their respective colors.

6.9 Component Group Filtering

Functions

- `template<typename L >`
`object_groups< L >` [scribo::filter::object_groups_mean_width](#) (const `object_groups< L >` &groups, **float** width)
- `template<typename L >`
`object_groups< L >` [scribo::filter::object_groups_size_ratio](#) (const `object_groups< L >` &groups, **float** max_size_ratio, **float** max_invalid_ratio_per_group)
- `template<typename L >`
`object_groups< L >` [scribo::filter::object_groups_small](#) (const `object_groups< L >` &groups, **unsigned** n_links)
- `template<typename L >`
`object_groups< L >` [scribo::filter::object_groups_with_holes](#) (const `object_groups< L >` &components, **unsigned** min_size)

6.9.1 Detailed Description

Component groups filtering routines.

6.9.2 Function Documentation

6.9.2.1 `template<typename L > object_groups<L> scribo::filter::object_groups_mean_width (const object_groups< L > &groups, float width)`

Filter groups having their object mean width too low.

Parameters

<i>in</i>	<i>groups</i>	Object group information.
<i>in</i>	<i>width</i>	Object group mean width must be greater or equal to this value.

Returns

Filtered object group information.

6.9.2.2 `template<typename L > object_groups<L> scribo::filter::object_groups_size_ratio (const object_groups< L > &groups, float max_size_ratio, float max_invalid_ratio_per_group)`

Invalidates groups with too much thin and high components.

For each components in the group, it computes the height/width ratio. If it is higher or equal to `max_size_ratio`, the component is counted as invalid.

If there are `max_invalid_ratio_per_group` invalid components, the group is invalidated.

Returns

An `object_group` structure potentially with invalidated groups.

6.9.2.3 `template<typename L > object_groups<L> scribo::filter::object_groups_small (const object_groups< L > &groups, unsigned n_links)`

Invalidate groups with few components.

Parameters

in	<i>groups</i>	Information about object groups.
in	<i>n_links</i>	The minimum number of links per group.

Returns

A copy of object group in which small groups have been invalidated.

6.9.2.4 `template<typename L > object_groups<L> scribo::filter::object_groups_with_holes (const object_groups< L > & components, unsigned min_size) [inline]`

Remove groups not having at least two background components of `min_size` pixels.

In order to verify the property of "having two background components", a component group must have at least one component with a hole.

Example: the letter 'o' has two background components: outside and inside the letter.

6.10 Component Link Filtering

Functions

- `template<typename L >`
`object_links< L >` `scribo::filter::object_links_aligned` (const `object_links< L >` &links, **float** max_alpha, anchor::Type anchor)
- `template<typename L >`
`object_links< L >` `scribo::filter::object_links_bbox_h_ratio` (const `object_links< L >` &links, **float** max_h_ratio)
- `template<typename L >`
`object_links< L >` `scribo::filter::object_links_bbox_overlap` (const `object_links< L >` &links, **float** max_overlap_ratio)
- `template<typename L >`
`object_links< L >` `scribo::filter::object_links_bbox_ratio` (const `object_links< L >` &links, **unsigned** dim, **float** max_ratio)
- `template<typename L >`
`object_links< L >` `scribo::filter::object_links_bbox_w_ratio` (const `object_links< L >` &links, **float** max_w_ratio)
- `template<typename L >`
`object_links< L >` `scribo::filter::object_links_bottom_aligned` (const `object_links< L >` &links, **float** max_alpha)
- `template<typename L >`
`object_links< L >` `scribo::filter::object_links_center_aligned` (const `object_links< L >` &links, **float** max_alpha)
- `template<typename L >`
`object_links< L >` `scribo::filter::object_links_left_aligned` (const `object_links< L >` &links, **float** max_alpha)
- `template<typename L >`
`object_links< L >` `scribo::filter::object_links_non_aligned_simple` (const `object_links< L >` &links, anchor::Type anchor, **float** max_alpha)
- `template<typename L >`
`object_links< L >` `scribo::filter::object_links_right_aligned` (const `object_links< L >` &links, **float** max_alpha)
- `template<typename L >`
`object_links< L >` `scribo::filter::object_links_top_aligned` (const `object_links< L >` &links, **float** max_alpha)

6.10.1 Detailed Description

Component links filtering routines.

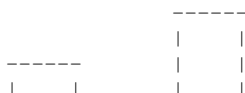
6.10.2 Function Documentation

6.10.2.1 `template<typename L > object_links<L> scribo::filter::object_links_aligned (const object_links< L > & links, float max_alpha, anchor::Type anchor)`

Invalidate links between two components according to a specific anchor.

Parameters

in	<i>links</i>	Object links information.
in	<i>max_alpha</i>	Maximum angle value (degrees).
in	<i>anchor</i>	Anchor used to compute angles.



Returns

A filtered object link information.

6.10.2.5 `template<typename L> object.links<L> scribo::filter::object_links_bbox_w_ratio (const object.links< L > & links, float max_w_ratio)`

Invalidate links between two objects with too different width.

Parameters

in	<i>links</i>	Link objects information.
in	<i>max_w_ratio</i>	The minimum width ratio of two linked bounding boxes.

Returns

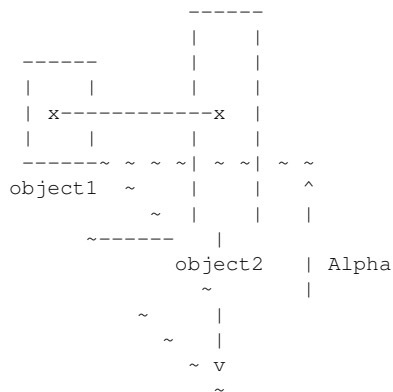
A filtered object link information.

6.10.2.6 `template<typename L> object.links<L> scribo::filter::object_links_bottom_aligned (const object.links< L > & links, float max_alpha)`

Invalidate links between two components if their bottom are not aligned.

Parameters

in	<i>links</i>	Object links information.
in	<i>max_alpha</i>	Maximum angle value (degrees).



The angle between the two bottoms must be lower than `max_alpha`.

6.10.2.7 `template<typename L> object.links<L> scribo::filter::object_links_center_aligned (const object.links< L > & links, float max_alpha)`

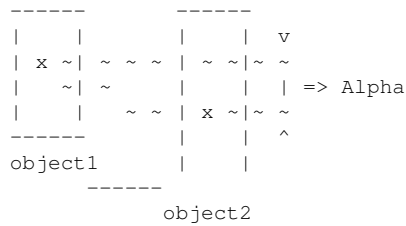
Invalidate links between two objects if their center are not aligned.

Parameters

in	<i>links</i>	Object links information.
in	<i>max_alpha</i>	Maximum angle value (degrees).

Returns

New link data.



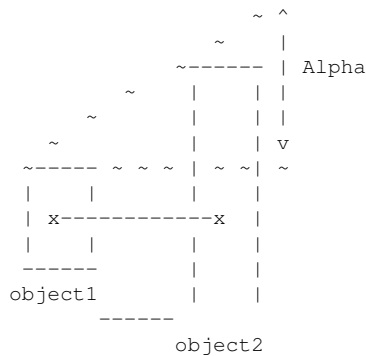
The angle between the two bottoms must be lower than `max_alpha`.

6.10.2.8 `template<typename L> object_links<L> scribo::filter::object_links_left_aligned (const object_links< L > & links, float max_alpha)`

Invalidate links between two objects if their left are not aligned.

Parameters

in	<i>links</i>	Object links information.
in	<i>max_alpha</i>	Maximum angle value (degrees).



The angle between the two lefts must be lower than `max_alpha`.

6.10.2.9 `template<typename L> object_links<L> scribo::filter::object_links_non_aligned_simple (const object_links< L > & links, anchor::Type anchor, float max_alpha)`

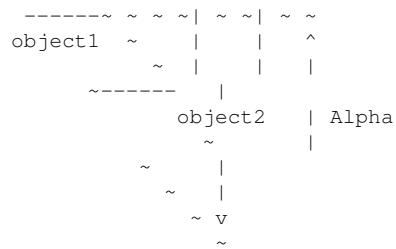
Invalidate links between two non aligned components. Alignment is based on a given anchor of object bounding boxes.

Parameters

in	<i>links</i>	Object links information.
in	<i>anchor</i>	Anchor from where the links are made.
in	<i>max_alpha</i>	Maximum angle value (degrees).

Exemple with `anchor == 1` (bottom horizontal filter):





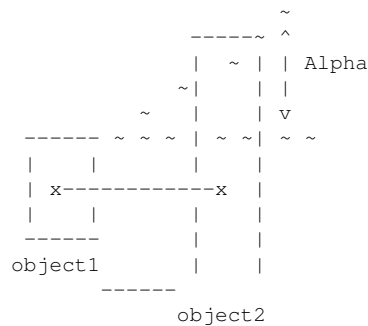
The angle between the two bottoms must be lower than `alpha`.

6.10.2.10 `template<typename L> object_links<L> scribo::filter::object_links_right_aligned (const object_links< L > & links, float max_alpha)`

Invalidate links between two objects if their right are not aligned.

Parameters

in	<i>links</i>	Object links information.
in	<i>max_alpha</i>	Maximum angle value (degrees).



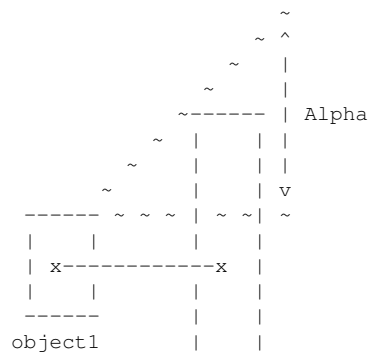
The angle between the two rights must be lower than `max_alpha`.

6.10.2.11 `template<typename L> object_links<L> scribo::filter::object_links_top_aligned (const object_links< L > & links, float max_alpha)`

Invalidate links between two objects if their top are not aligned.

Parameters

in	<i>links</i>	Object links information.
in	<i>max_alpha</i>	Maximum angle value (degrees).



`object2`

The angle between the two tops must be lower than `max_alpha`.

6.11 Component Filtering

Functions

- `template<typename I, typename N, typename V >`
mln::trait::concrete< I >::ret [scribo::filter::components_large](#) (const **Image**< I > &input_, const **Neighborhood**< N > &nbh_, const V &label_type, **unsigned** max_size)
- `template<typename L >`
component_set< L > [scribo::filter::components_large](#) (const **component_set**< L > &components, **unsigned** max_size)
- `template<typename L >`
component_set< L > [scribo::filter::components_on_border](#) (const **component_set**< L > &components)
- `template<typename I, typename N, typename V >`
mln::trait::concrete< I >::ret [scribo::filter::components_small](#) (const **Image**< I > &input_, const **Neighborhood**< N > &nbh_, V &nlabels, **unsigned** min_size)
- `template<typename L >`
component_set< L > [scribo::filter::components_small](#) (const **component_set**< L > &components, **unsigned** min_size)
- `template<typename I, typename N, typename V >`
mln::trait::concrete< I >::ret [scribo::filter::components_thin](#) (const **Image**< I > &input_, const **Neighborhood**< N > &nbh_, const V &label_type, **unsigned** min_thickness)
- `template<typename L >`
component_set< L > [scribo::filter::components_thin](#) (const **component_set**< L > &components, **unsigned** min_thickness)
- `template<typename L >`
component_set< L > [scribo::filter::components_v_thin](#) (const **component_set**< L > &comps, **unsigned** min_thinness)
- `template<typename L >`
component_set< L > [scribo::filter::components_with_two_holes](#) (const **component_set**< L > &components, **unsigned** min_size)
- `template<typename I, typename N, typename V >`
mln::trait::concrete< I >::ret [scribo::filter::objects_h_thick](#) (const **Image**< I > &input_, const **Neighborhood**< N > &nbh_, const V &label_type, **unsigned** max_thickness)
- `template<typename L >`
component_set< L > [scribo::filter::objects_h_thick](#) (const **component_set**< L > &comps, **unsigned** max_thickness)
- `template<typename I, typename N, typename V >`
mln::trait::concrete< I >::ret [scribo::filter::objects_h_thin](#) (const **Image**< I > &input_, const **Neighborhood**< N > &nbh_, const V &label_type, **unsigned** min_thinness)
- `template<typename L >`
component_set< L > [scribo::filter::objects_h_thin](#) (const **component_set**< L > &comps, **unsigned** min_thinness)
- `template<typename L >`
component_set< L > [scribo::filter::objects_size_ratio](#) (const **component_set**< L > &comps, **float** min_size_ratio)
- `template<typename I, typename N, typename V >`
mln::trait::concrete< I >::ret [scribo::filter::objects_thick](#) (const **Image**< I > &input_, const **Neighborhood**< N > &nbh_, const V &label_type, **unsigned** max_thickness)
- `template<typename L >`
component_set< L > [scribo::filter::objects_thick](#) (const **component_set**< L > &components, **unsigned** max_thickness)
- `template<typename I, typename N, typename V >`
mln::trait::concrete< I >::ret [scribo::filter::objects_v_thick](#) (const **Image**< I > &input_, const **Neighborhood**< N > &nbh_, const V &label_type, **unsigned** max_thickness)
- `template<typename L >`
component_set< L > [scribo::filter::objects_v_thick](#) (const **component_set**< L > &comps, **unsigned** max_thickness)

- `template<typename I, typename N, typename V >`
`mln::trait::concrete< I >::ret scribo::filter::objects_v_thin` (const `Image< I >` &input, const `Neighborhood< N >` &nbh, const `V` &label_type, **unsigned** min_thinness)
- `template<typename L >`
`component_set< L >` `scribo::filter::objects_with_holes` (const `component_set< L >` &components, **unsigned** min_holes_count, **unsigned** min_size)

6.11.1 Detailed Description

Component filtering routines.

6.11.2 Function Documentation

6.11.2.1 `template<typename I, typename N, typename V > mln::trait::concrete< I >::ret scribo::filter::components_large` (const `Image< I >` & *input_*, const `Neighborhood< N >` & *nbh_*, const `V` & *label_type*, **unsigned** *max_size*)

Remove large objects in a binary image.

Set to 'false' all the removed objects.

Parameters

in	<i>input_</i>	A binary image.
in	<i>nbh_</i>	A neighborhood used for labeling <i>input_</i> .
in	<i>label_type</i>	The label type used for labeling.
in	<i>max_size</i>	The minimum cardinality of an object.

Returns

A binary image without large objects.

6.11.2.2 `template<typename L > component_set<L> scribo::filter::components_large` (const `component_set< L >` & *components*, **unsigned** *max_size*) `[inline]`

Remove too large components.

Parameters

in	<i>components</i>	An object image.
in	<i>max_size</i>	The maximum cardinality of an object.

Returns

A component set with large components set to `component::Ignored`.

6.11.2.3 `template<typename L > component_set<L> scribo::filter::components_on_border` (const `component_set< L >` & *components*) `[inline]`

Remove components located on image borders.

Parameters

in	<i>components</i>	An object image.
----	-------------------	------------------

Returns

A component set with large components set to [component::Ignored](#).

6.11.2.4 `template<typename I, typename N, typename V> mIn::trait::concrete<I>::ret scribo::filter::components_small (const Image<I> & input, const Neighborhood<N> & nbh, V & nlabels, unsigned min_size)`

Remove small components in a binary image.

Set to 'false' all the removed components.

Parameters

in	<i>input</i>	A binary image.
in	<i>nbh</i>	A neighborhood used for labeling <i>input_</i> .
out	<i>nlabels</i>	Return the number of components. Defines also the type used for labeling.
in	<i>min_size</i>	The minimum cardinality of an object.

Returns

A binary image without small components.

6.11.2.5 `template<typename L> component_set<L> scribo::filter::components_small (const component_set<L> & components, unsigned min_size)`

Remove too small components.

Parameters

in	<i>components</i>	An object image.
in	<i>min_size</i>	The minimum cardinality of an object.

Returns

A component set with small components set to [component::Ignored](#).

6.11.2.6 `template<typename I, typename N, typename V> mIn::trait::concrete<I>::ret scribo::filter::components_thin (const Image<I> & input_, const Neighborhood<N> & nbh_, const V & label_type, unsigned min_thickness) [inline]`

Remove components thinner or equal to *min_thickness*.

Parameters

in	<i>input_</i>	a binary image.
in	<i>nbh_</i>	a neighborhood used in labeling algorithms.
in	<i>label_type</i>	the label type used for labeling.
in	<i>min_thickness</i>	the minimum thickness value.

Returns

A binary image without thin components.

6.11.2.7 `template<typename L> component_set<L> scribo::filter::components_thin (const component_set< L > & components, unsigned min_thickness) [inline]`

Remove lines of text thinner or equal to `min_thickness`.

Parameters

in	<i>components</i>	An object image.
in	<i>min_thickness</i>	the minimum thickness value.

Returns

An object image without too thin components.

6.11.2.8 `template<typename L> component_set<L> scribo::filter::components_v_thin (const component_set< L > & comps, unsigned min_thinness) [inline]`

Remove lines of text thinner or equal to `min_thinness`.

Parameters

in	<i>comps</i>	A component set.
in	<i>min_thinness</i>	the minimum thinness value.

Returns

An object image without too thin vertical components.

6.11.2.9 `template<typename L> component_set<L> scribo::filter::components_with_two_holes (const component_set< L > & components, unsigned min_size) [inline]`

Remove components having at least two holes.

This is a fastest version since it is optimized for 2 holes detection.

Parameters

in	<i>components</i>	A component set.
in	<i>min_size</i>	The minimum hole area to take a hole into account.

Returns

A component where the component having at least two holes are invalidated.

6.11.2.10 `template<typename I, typename N, typename V> mln::trait::concrete< I >::ret scribo::filter::objects_h_thick (const Image< I > & input_, const Neighborhood< N > & nbh_, const V & label_type, unsigned max_thickness) [inline]`

Remove objects horizontally thicker or equal to `max_thickness`.

Parameters

in	<i>input_</i>	A binary image.
in	<i>nbh_</i>	A neighborhood used in labeling algorithms.
in	<i>label_type</i>	The label type used for labeling.
in	<i>max_thickness</i>	The maximum thickness value.

Returns

A binary image without thick objects.

6.11.2.11 `template<typename L> component_set<L> scribo::filter::objects_h.thick (const component_set< L> & comps, unsigned max_thickness) [inline]`

Remove objects horizontally thicker or equal to `max_thickness`.

Parameters

in	<i>comps</i>	Component data.
in	<i>max_thickness</i>	The minimum thickness value.

Returns

A component data set without too thick components.

6.11.2.12 `template<typename I, typename N, typename V> mln::trait::concrete< I>::ret scribo::filter::objects_h.thin (const Image< I> & input_, const Neighborhood< N> & nbh_, const V & label_type, unsigned min_thinness) [inline]`

Remove components thinner or equal to `min_thinness`.

Parameters

in	<i>input_</i>	a binary image.
in	<i>nbh_</i>	a neighborhood used in labeling algorithms.
in	<i>label_type</i>	the label type used for labeling.
in	<i>min_thinness</i>	the minimum thinness value.

Returns

A binary image without `h_thin` components.

6.11.2.13 `template<typename L> component_set<L> scribo::filter::objects_h.thin (const component_set< L> & comps, unsigned min_thinness) [inline]`

Remove lines of text thinner or equal to `min_thinness`.

Parameters

in	<i>comps</i>	A component set.
in	<i>min_thinness</i>	the minimum thinness value.

Returns

An object image without too thin vertical components.

6.11.2.14 `template<typename L> component_set<L> scribo::filter::objects_size_ratio (const component_set< L> & comps, float min_size_ratio)`

Invalidate components with a height/width ratio too low.

Compute the ratio height/width from the component bounding boxes and compare it to `size_ratio`.

If the height/width ratio is lower than `min_size_ratio` then the component is invalidated.

```
6.11.2.15  template<typename I , typename N , typename V > mIn::trait::concrete< I >::ret scribo::filter::objects_thick
            ( const Image< I > & input_, const Neighborhood< N > & nbh_, const V & label_type, unsigned
              max_thickness ) [inline]
```

Remove components thicker or equal to `max_thickness`.

Parameters

in	<i>input_</i>	A binary image.
in	<i>nbh_</i>	A neighborhood used in labeling algorithms.
in	<i>label_type</i>	The label type used for labeling.
in	<i>max_thickness</i>	The maximum thickness value.

Returns

A binary image without thick components.

```
6.11.2.16  template<typename L > component_set<L> scribo::filter::objects_thick ( const component_set< L > &
            components, unsigned max_thickness ) [inline]
```

Remove components thicker or equal to `max_thickness`.

Parameters

in	<i>components</i>	An object image.
in	<i>max_thickness</i>	The maximum thickness value.

Returns

An object image without too thick components.

```
6.11.2.17  template<typename I , typename N , typename V > mIn::trait::concrete< I >::ret scribo::filter::objects_v_thick
            ( const Image< I > & input_, const Neighborhood< N > & nbh_, const V & label_type, unsigned
              max_thickness ) [inline]
```

Remove components vertically thicker or equal to `max_thickness`.

Parameters

in	<i>input_</i>	A binary image.
in	<i>nbh_</i>	A neighborhood used in labeling algorithms.
in	<i>label_type</i>	The label type used for labeling.
in	<i>max_thickness</i>	The maximum thickness value.

Returns

A binary image without thick components.

6.11.2.18 `template<typename L> component_set<L> scribo::filter::objects_v.thick (const component_set< L > & comps, unsigned max_thickness) [inline]`

Remove components vertically thicker or equal to `max_thickness`.

Parameters

<code>in</code>	<code>comps</code>	A component set.
<code>in</code>	<code>max_thickness</code>	The maximum thickness value.

Returns

An object image without too thick components.

6.11.2.19 `template<typename I, typename N, typename V> mln::trait::concrete< I >::ret scribo::filter::objects_v.thin (const Image< I > & input, const Neighborhood< N > & nbh, const V & label_type, unsigned min_thinness) [inline]`

Remove components thinner or equal to `min_thinness`.

Parameters

<code>in</code>	<code>input</code>	a binary image.
<code>in</code>	<code>nbh</code>	a neighborhood used in labeling algorithms.
<code>in</code>	<code>label_type</code>	the label type used for labeling.
<code>in</code>	<code>min_thinness</code>	the minimum thinness value.

Returns

A binary image without `v_thin` components.

6.11.2.20 `template<typename L> component_set<L> scribo::filter::objects_with_holes (const component_set< L > & components, unsigned min_holes_count, unsigned min_size)`

Remove components having a minimum number of holes.

Parameters

<code>in</code>	<code>components</code>	A component set.
<code>in</code>	<code>min_holes_count</code>	If a component have at least <code>min_holes_count</code> holes it is invalidated.
<code>in</code>	<code>min_size</code>	The minimum hole area to take a hole into account.

Returns

A component where the component having too much holes are invalidated.

6.12 Element Filtering

Functions

- `template<typename L >`
void `scribo::filter::images_in_paragraph` (document< L > &doc)
- `template<typename L >`
void `scribo::filter::objects_in_borders` (component_set< L > &components, **float** vratio, **float** hratio)
- `template<typename L >`
void `scribo::filter::separators_in_borders` (document< L > &doc, **float** vratio, **float** hratio)
- `template<typename L >`
void `scribo::filter::separators_in_element` (document< L > &doc)
- `template<typename L >`
void `scribo::filter::separators_in_paragraph` (document< L > &doc, **unsigned** hmin_size, **unsigned** vmin_size)
- `template<typename L >`
void `scribo::filter::separators_vert_in_borders` (document< L > &doc)

6.12.1 Detailed Description

Elements filtering routines.

6.12.2 Function Documentation

6.12.2.1 `template<typename L > void scribo::filter::images_in_paragraph (document< L > & doc)`

Invalidate false positive images.

Parameters

<code>in, out</code>	<code>doc</code>	A document structure.
----------------------	------------------	-----------------------

6.12.2.2 `template<typename L > void scribo::filter::objects_in_borders (component_set< L > & components, float vratio, float hratio)`

Invalidate components located close to the image borders.

Parameters

<code>in, out</code>	<code>components</code>	A component set.
<code>in</code>	<code>vratio</code>	Ratio to be used for evaluating the inner border size in which vertical separators will be invalidated.
<code>in</code>	<code>hratio</code>	Ratio to be used for evaluating the inner border size in which horizontal separators will be invalidated.

Warning

It only invalidates components in the given `component_set`.

```

-----
|_!____!_|
|!____!|<----- Components located in this area are
|!____!|invalidated.
|!____!|
|_!____!_|
|!____!|

```


6.12.2.6 `template<typename L> void scribo::filter::separators_vert_in_borders (document< L > & doc)`

Invalidate vertical separators located close to the image borders.

Parameters

<code>in, out</code>	<code>doc</code>	A document structure.
----------------------	------------------	-----------------------

Warning: it does not remove separators from separator image. **It** only invalidate separator components in their respective [component_set](#).

```

-----
|_!____!_|
|!      !| <----- Separators located in this area are
|!      !|      invalidated.
|!      !|
|_!____!_|
|!      !|
|!      !|
-----

```

6.13 Element Filtering

Modules

- [Component Filtering](#)
- [Component Group Filtering](#)
- [Component Link Filtering](#)
- [Element Filtering](#)
- [Line Link Filtering](#)
- [Paragraph Filtering](#)

6.13.1 Detailed Description

Element filtering routines.

6.14 Line Link Filtering

Functions

- `template<typename L >`
`line_links< L > scribo::filter::line_links_x_height (const line_links< L > &links)`

6.14.1 Detailed Description

Line link filtering routines.

6.14.2 Function Documentation

6.14.2.1 `template<typename L > line_links<L> scribo::filter::line_links_x_height (const line_links< L > & links)`

Filter line links according to character x height.

Parameters

<i>in</i>	<i>links</i>	Line links information.
-----------	--------------	-------------------------

Returns

A new [line_links](#) structure where some links may have been invalidated.

6.15 Paragraph Filtering

Functions

- `template<typename L >`
`paragraph_set< L >` [scribo::filter::paragraphs_bbox_overlap](#) (const `paragraph_set< L >` &parset)
- `template<typename L >`
`void` [scribo::filter::paragraphs_in_borders](#) (document< L > &doc)
- `template<typename L >`
`void` [scribo::filter::paragraphs_in_image](#) (document< L > &doc)

6.15.1 Detailed Description

Paragraph filtering routines.

6.15.2 Function Documentation

6.15.2.1 `template<typename L > paragraph_set<L> scribo::filter::paragraphs_bbox_overlap (const paragraph_set< L > &parset)`

Remove invalid paragraphs.

Parameters

<i>in</i>	<i>parset</i>	A paragraph set.
-----------	---------------	------------------

Returns

A paragraph set with invalid paragraphs tag set to Paragraph::Ignored.

6.15.2.2 `template<typename L > void scribo::filter::paragraphs_in_borders (document< L > &doc)`

Invalidate paragraphs located close to the image borders.

Parameters

<i>in, out</i>	<i>doc</i>	A document structure.
----------------	------------	-----------------------

Warning: it does not remove paragraphs from separator image. It only invalidate separator components in their respective [component_set](#).

```

-----
|_!____!_|
|!      !|<----- Paragraphs located in this area are
|!      !|      invalidated.
|!      !|
|_!____!_|
|!      !|
-----

```

6.15.2.3 `template<typename L > void scribo::filter::paragraphs_in_image (document< L > &doc)`

Remove invalid paragraphs.

Parameters

<i>in, out</i>	<i>doc</i>	A document structure.
----------------	------------	-----------------------

Returns

A paragraph set with invalid paragraphs tag set to Paragraph::Ignored.

6.16 Layout Analysis

Functions

- `template<typename I >`
`mln::util::array< mln_box(I)> scribo::layout::xy_cut (const Image< I > &ima, int min_height, int min_width)`

6.16.1 Detailed Description

Document Layout Analysis algorithms.

6.16.2 Function Documentation

6.16.2.1 `template<typename I > mln::util::array<mln_box(I)> scribo::layout::xy_cut (const Image< I > & ima, int min_height, int min_width)`

XY-Cut layout analysis algorithm.

This algorithm is an implementation inspired by [4], [2] and [3].

It recursively subdivides empty spaces in the document until a minimum division size is reached. The latter is defined with `min_height` and `min_width`.

Parameters

<code>in</code>	<code>ima</code>	A binary image.
<code>in</code>	<code>min_height</code>	The minimum height of a subdivision.
<code>in</code>	<code>min_width</code>	The minimum width of a subdivision.

Returns

An array of component group bounding boxes.

6.17 Components Extraction

Functions

- template<typename I , typename J , typename N , typename V >
 component_set< typename
mln::trait::ch_value< I, V >
 ::ret > **scribo::primitive::extract::components** (const **Image**< I > &input, const **Image**< J > &binary_input,
 const **Neighborhood**< N > &nbh, V &ncomponents, component::Type type=component::Undefined)
- template<typename I , typename N , typename V >
 component_set< typename
mln::trait::ch_value< I, V >
 ::ret > **scribo::primitive::extract::components** (const **Image**< I > &binary_input, const **Neighborhood**< N >
 &nbh, V &ncomponents, component::Type type=component::Undefined)

6.17.1 Detailed Description

All routines/algorithms to extract components.

6.17.2 Function Documentation

6.17.2.1 `template<typename I , typename J , typename N , typename V > component_set<typename mln::trait::ch_value< I, V >::ret> scribo::primitive::extract::components (const Image< I > & input, const Image< J > & binary_input, const Neighborhood< N > & nbh, V & ncomponents, component::Type type = component::Undefined) [inline]`

Extract components in a binary image.

Parameters

in	<i>input</i>	A RGB image.
in	<i>binary_input</i>	A binary image. Components must be set to 'True'. and background to 'false'.
in	<i>nbh</i>	A neighborhood to be used for labeling.
in, out	<i>ncomponents</i>	Will store the numbers of components found.
in	<i>type</i>	The default component type set to components.

Returns

An image of labeled components.

6.17.2.2 `template<typename I , typename N , typename V > component_set<typename mln::trait::ch_value< I, V >::ret> scribo::primitive::extract::components (const Image< I > & binary_input, const Neighborhood< N > & nbh, V & ncomponents, component::Type type = component::Undefined) [inline]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

Parameters

in	<i>binary_input</i>	A binary image. Components must be set to 'True'. and background to 'false'.
in	<i>nbh</i>	A neighborhood to be used for labeling.
in, out	<i>ncomponents</i>	Will store the numbers of components found.
in	<i>type</i>	The default component type set to components.

Returns

A [component_set](#).

6.18 Primitive Extraction

Modules

- [Lines and Separators extraction](#)

6.18.1 Detailed Description

All routines/algorithms to extract primitive objects.

6.19 Lines and Separators extraction

Functions

- `template<typename I >`
`mln::trait::concrete< I >::ret scribo::primitive::extract::horizontal_separators` (const `Image< I >` &input, unsigned line_length)
- `template<typename I, typename N, typename V, typename W >`
`component_set< typename`
`mln::trait::ch_value< I, V >`
`::ret > scribo::primitive::extract::lines_discontinued` (const `Image< I >` &input_, const `Neighborhood< N >` &nbh_, V &nlines, const `Window< W >` &win_, unsigned rank_k)
- `template<typename I, typename N, typename V >`
`component_set< typename`
`mln::trait::ch_value< I, V >`
`::ret > scribo::primitive::extract::lines_h_discontinued` (const `Image< I >` &input, const `Neighborhood< N >` &nbh, V &nlines, unsigned line_length, unsigned rank_k)
- `template<typename I >`
`mln::trait::concrete< I >::ret scribo::primitive::extract::lines_h_pattern` (const `Image< I >` &input, unsigned length, unsigned delta)
- `template<typename I, typename N, typename V >`
`component_set< typename`
`mln::trait::ch_value< I, V >`
`::ret > scribo::primitive::extract::lines_h_single` (const `Image< I >` &input, const `Neighborhood< N >` &nbh, const V &nlines, unsigned min_line_length, float w_h_ratio)
- `template<typename L >`
`component_set< L > scribo::primitive::extract::lines_h_single` (const `component_set< L >` &components, unsigned min_line_length, float w_h_ratio)
- `template<typename I, typename N, typename V >`
`component_set< typename`
`mln::trait::ch_value< I, V >`
`::ret > scribo::primitive::extract::lines_h_thick_and_single` (const `Image< I >` &input, const `Neighborhood< N >` &nbh, V &nlines, unsigned min_line_length, float h_w_ratio)
- `template<typename I >`
`mln::trait::concrete< I >::ret scribo::primitive::extract::lines_h_thick_and_thin` (const `Image< I >` &binary_image, unsigned length, unsigned delta, float p_few=0.2, float p_enough=0.6, float ratio=8)
- `template<typename I, typename W >`
`mln::trait::concrete< I >::ret scribo::primitive::extract::lines_pattern` (const `Image< I >` &input_, unsigned length, unsigned dir, const `Window< W >` &win_)
- `template<typename I, typename N, typename V, typename W >`
`component_set< typename`
`mln::trait::ch_value< I, V >`
`::ret > scribo::primitive::extract::lines_thick` (const `Image< I >` &input_, const `Neighborhood< N >` &nbh_, V &nlines, unsigned line_length)
- `template<typename I, typename N, typename V, typename W >`
`component_set< typename`
`mln::trait::ch_value< I, V >`
`::ret > scribo::primitive::extract::lines_thick` (const `Image< I >` &input_, const `Neighborhood< N >` &nbh_, V &nlines, const `Window< W >` &win_)
- `template<typename I, typename N, typename V >`
`component_set< typename`
`mln::trait::ch_value< I, V >`
`::ret > scribo::primitive::extract::lines_v_discontinued` (const `Image< I >` &input, const `Neighborhood< N >` &nbh, V &nlines, unsigned line_length, unsigned rank_k)
- `template<typename I >`
`mln::trait::concrete< I >::ret scribo::primitive::extract::lines_v_pattern` (const `Image< I >` &input, unsigned length, unsigned delta)

- `template<typename I , typename N , typename V >`
`component_set< typename`
`mln::trait::ch_value< I, V >`
`::ret > scribo::primitive::extract::lines_v_single (const Image< I > &input, const Neighborhood< N > &nbh,`
`const V &nlines, unsigned min_line_length, float h_w_ratio)`
- `template<typename L >`
`component_set< L > scribo::primitive::extract::lines_v_single (const component_set< L > &components,`
`unsigned min_line_length, float h_w_ratio)`
- `template<typename I , typename N , typename V >`
`component_set< typename`
`mln::trait::ch_value< I, V >`
`::ret > scribo::primitive::extract::lines_v_thick_and_single (const Image< I > &input, const Neighborhood<`
`N > &nbh, V &nlines, unsigned min_line_length, float h_w_ratio)`
- `template<typename I >`
`mln::trait::concrete< I >::ret scribo::primitive::extract::separators (const Image< I > &input, unsigned`
`line_length)`
- `template<typename I >`
`mln::trait::concrete< I >::ret scribo::primitive::extract::separators_nonvisible (const Image< I > &in_)`
- `template<typename I >`
`mln::trait::concrete< I >::ret scribo::primitive::extract::vertical_separators (const Image< I > &input, un-`
`signed line_length)`

6.19.1 Detailed Description

All routines/algorithms to extract lines and separators.

6.19.2 Function Documentation

6.19.2.1 `template<typename I > mln::trait::concrete< I >::ret scribo::primitive::extract::horizontal_separators (const Image< I > & input, unsigned line_length)`

Extract horizontal separators.

Parameters

<code>in</code>	<code>input</code>	A binary image.
<code>in</code>	<code>line_length</code>	The minimum line length.

Returns

A binary image where separators are set to 'True'.

See Also

`primitive::remove::separators`

6.19.2.2 `template<typename I , typename N , typename V , typename W > component_set<typename mln::trait::ch_value< I, V >::ret> scribo::primitive::extract::lines_discontinued (const Image< I > & input_, const Neighborhood< N > & nbh_, V & nlines, const Window< W > & win_, unsigned rank_k)`

Extract discontinued lines in a binary image.

Based on a rank filter.

Parameters

in	<i>input_</i>	A binary image.
in	<i>nbh_</i>	The neighborhood used for labeling image the lines.
in, out	<i>nlines</i>	The label type used for labeling.
in	<i>win_</i>	A Window used to extract lines.
in	<i>rank_k</i>	Rank used for filtering.

Returns

An image in which lines are labeled.

Precondition

`win.length() > 2 * (rank_k + 1)`

6.19.2.3 `template<typename I , typename N , typename V > component_set<typename mIn::trait::ch_value< I , V >::ret> scribo::primitive::extract::lines_h_discontinued (const Image< I > & input, const Neighborhood< N > & nbh, V & nlines, unsigned line_length, unsigned rank_k)`

Extract horizontal discontinued lines.

Parameters

in	<i>input</i>	A binary image.
in	<i>nbh</i>	A neighborhood used to label lines.
in, out	<i>nlines</i>	The number of lines found.
in	<i>line_length</i>	The minimum line length expected. (must be odd).
in	<i>rank_k</i>	Rank filter parameter.

Returns

An image in which lines are labeled with a value different from 0.

6.19.2.4 `template<typename I > mIn::trait::concrete< I >::ret scribo::primitive::extract::lines_h_pattern (const Image< I > & input, unsigned length, unsigned delta)`

Extract horizontal lines matching a specific pattern.

Parameters

in	<i>input</i>	A binary image.
in	<i>length</i>	The minimum line length.
in	<i>delta</i>	Distance between the object pixel and the background pixel.

Returns

An image of horizontal lines.

```

o
|   ^
|   | Delta
|   v
X
|   ^
|   | Delta
|   v
o

```

Using a delta of 0 is equivalent to the use of a c2_row neighborhood.

```
6.19.2.5  template<typename I , typename N , typename V > component_set<typename mIn::trait::ch_value< I , V
>::ret> scribo::primitive::extract::lines_h_single ( const Image< I > & input, const Neighborhood< N > & nbh,
const V & nlines, unsigned min_line_length, float w_h_ratio )
```

Fast Extraction of single horizontal thick lines.

Only single non discontinued lines are correctly extracted with this routine.

Parameters

in	input	A binary image.
in	nbh	The neighborhood used for labeling image components.
in, out	nlines	Type used for labeling.
in	min_line_length	The minimum line length.
in	w_h_ratio	The minimum ratio width/height object bounding boxes to consider an object as a single line.

Returns

An image in which only horizontal single lines are labeled.

```
6.19.2.6  template<typename L > component_set<L> scribo::primitive::extract::lines_h_single ( const component_set< L >
& components, unsigned min_line_length, float w_h_ratio )
```

Fast Extraction of single horizontal thick lines.

Only single non discontinued lines are correctly extracted with this routine.

Parameters

in	components	A labeled image.
in	min_line_length	The minimum line length.
in	w_h_ratio	The minimum ratio width/height object bounding boxes to consider an object as a single line.

Returns

An image in which only horizontal single lines are labeled.

```
6.19.2.7  template<typename I , typename N , typename V > component_set<typename mIn::trait::ch_value< I , V
>::ret> scribo::primitive::extract::lines_h.thick_and_single ( const Image< I > & input, const Neighborhood<
N > & nbh, V & nlines, unsigned min_line_length, float h_w_ratio )
```

Extract horizontal thick lines in a binary image.

Only non discontinued lines are correctly extracted with this routine. Only lines matching the given criterions are kept in the result.

Parameters

in	input	A binary image.
in	nbh	The neighborhood used for labeling image components.
in, out	nlines	Type used for labeling.
in	min_line_length	The minimum line length.
in	h_w_ratio	The minimum ratio width/height object bounding boxes to consider an object as a single line.

Returns

An image in which lines are labeled.

6.19.2.8 `template<typename I> mIn::trait::concrete<I>::ret scribo::primitive::extract::lines_h.thick_and_thin (const Image<I> & binary_image, unsigned length, unsigned delta, float p_few = 0.2, float p_enough = 0.6, float ratio = 8)`

Extract both thick and thin horizontal lines.

6.19.2.9 `template<typename I, typename W> mIn::trait::concrete<I>::ret scribo::primitive::extract::lines_pattern (const Image<I> & input_, unsigned length, unsigned dir, const Window<W> & win_)`

Extract lines with a specific pattern.

Parameters

in	<i>input_</i>	A binary image.
in	<i>length</i>	The minimum line length.
in	<i>dir</i>	The direction of the lines.
in	<i>win_</i>	A window corresponding to the line pattern.

Returns

A image with lines of direction *dir*.

6.19.2.10 `template<typename I, typename N, typename V, typename W> component_set<typename mIn::trait::ch_value<I, V>::ret> scribo::primitive::extract::lines_thick (const Image<I> & input_, const Neighborhood<N> & nbh_, V & nlines, unsigned line_length)`

Extract thick lines in a binary image.

Only non discontinued lines are correctly extracted with this routine.

Parameters

in	<i>input_</i>	A binary image.
in	<i>nbh_</i>	The neighborhood used for labeling image components.
in, out	<i>nlines</i>	Type used for labeling.
in	<i>line_length</i>	The minimum line length.

Returns

An image in which lines are labeled.

6.19.2.11 `template<typename I, typename N, typename V, typename W> component_set<typename mIn::trait::ch_value<I, V>::ret> scribo::primitive::extract::lines_thick (const Image<I> & input_, const Neighborhood<N> & nbh_, V & nlines, const Window<W> & win_)`

Extract thick lines in a binary image. Only non discontinued lines are correctly extracted with this routine.

Parameters

in	<i>input_</i>	A binary image.
in	<i>nbh_</i>	The neighborhood used for labeling image components.

<i>in, out</i>	<i>nlines</i>	Type used for labeling.
<i>in</i>	<i>win_</i>	Window used to extract the lines

Returns

An image in which lines are labeled.

6.19.2.12 `template<typename I , typename N , typename V > component_set<typename mIn::trait::ch_value< I , V >::ret> scribo::primitive::extract::lines_v_discontinued (const Image< I > & input, const Neighborhood< N > & nbh, V & nlines, unsigned line_length, unsigned rank_k)`

Extract vertical discontinued lines.

Parameters

<i>in</i>	<i>input</i>	A binary image.
<i>in</i>	<i>nbh</i>	A neighborhood used to label lines.
<i>in, out</i>	<i>nlines</i>	The number of lines found.
<i>in</i>	<i>line_length</i>	The minimum line length expected. (must be odd).
<i>in</i>	<i>rank_k</i>	Rank filter parameter.

Returns

An image in which lines are labeled with a value different from 0.

6.19.2.13 `template<typename I > mIn::trait::concrete< I >::ret scribo::primitive::extract::lines_v_pattern (const Image< I > & input, unsigned length, unsigned delta)`

Extract vertical lines matching a specific pattern.

Parameters

<i>in</i>	<i>input</i>	A binary image.
<i>in</i>	<i>length</i>	The minimum line length.
<i>in</i>	<i>delta</i>	space between the first background pixels and the line pixels (usually 2 or 3).

Returns

An image of vertical lines.

6.19.2.14 `template<typename I , typename N , typename V > component_set<typename mIn::trait::ch_value< I , V >::ret> scribo::primitive::extract::lines_v_single (const Image< I > & input, const Neighborhood< N > & nbh, const V & nlines, unsigned min_line_length, float h_w_ratio)`

Fast Extraction of single vertical thick lines.

Only single non discontinued lines are correctly extracted with this routine.

Parameters

<i>in</i>	<i>input</i>	A binary image.
<i>in</i>	<i>nbh</i>	The neighborhood used for labeling image components.
<i>in</i>	<i>nlines</i>	Type used for labeling.
<i>in</i>	<i>min_line_length</i>	The minimum line length.

<i>in</i>	<i>h_w_ratio</i>	The minimum ratio height/width object bounding boxes to consider an object as a single line.
-----------	------------------	--

Returns

An image in which only vertical single lines are labeled.

6.19.2.15 `template<typename L > component_set<L> scribo::primitive::extract::lines_v_single (const component_set< L > & components, unsigned min_line_length, float h_w_ratio)`

Fast Extraction of single vertical thick lines.

Only single non discontinued lines are correctly extracted with this routine.

Parameters

<i>in</i>	<i>components</i>	A labeled image.
<i>in</i>	<i>min_line_length</i>	The minimum line length.
<i>in</i>	<i>h_w_ratio</i>	The minimum ratio height/width object bounding boxes to consider an object as a single line.

Returns

An image in which only vertical single lines are labeled.

6.19.2.16 `template<typename I , typename N , typename V > component_set<typename mln::trait::ch_value< I , V >::ret> scribo::primitive::extract::lines_v_thick_and_single (const Image< I > & input, const Neighborhood< N > & nbh, V & nlines, unsigned min_line_length, float h_w_ratio)`

Extract vertical thick lines in a binary image.

Only non discontinued lines are correctly extracted with this routine. Only lines matching the given criterions are kept in the result.

Parameters

<i>in</i>	<i>input</i>	A binary image.
<i>in</i>	<i>nbh</i>	The neighborhood used for labeling image components.
<i>in, out</i>	<i>nlines</i>	Type used for labeling.
<i>in</i>	<i>min_line_length</i>	The minimum line length.
<i>in</i>	<i>h_w_ratio</i>	The minimum ratio height/width object bounding boxes to consider an object as a single line.

Returns

An image in which lines are labeled.

6.19.2.17 `template<typename I > mln::trait::concrete< I >::ret scribo::primitive::extract::separators (const Image< I > & input, unsigned line_length)`

Extract vertical and horizontal separators.

Parameters

<i>in</i>	<i>input</i>	A binary image.
<i>in</i>	<i>line_length</i>	The minimum line length.

Returns

A binary image were separators are set to 'True'.

See Also

`primitive::remove::separators`

6.19.2.18 `template<typename I> mIn::trait::concrete<I>::ret scribo::primitive::extract::separators_nonvisible (const Image<I> & in)`

Find non visible separators. Based on components alignments.

6.19.2.19 `template<typename I> mIn::trait::concrete<I>::ret scribo::primitive::extract::vertical_separators (const Image<I> & input, unsigned line_length)`

Extract vertical separators.

Parameters

<i>in</i>	<i>input</i>	A binary image.
<i>in</i>	<i>line_length</i>	The minimum line length.

Returns

A binary image were separators are set to 'True'.

See Also

`primitive::remove::separators`

6.20 Routines

Modules

- [Binarization](#)
- [Components Extraction](#)
- [Debug](#)
- [Element Filtering](#)
- [Layout Analysis](#)
- [Primitive Extraction](#)
- [Text Extraction](#)
- [Text Recognition](#)
- [Toolchains](#)

6.20.1 Detailed Description

All routines/algorithms provided in Scribo.

6.21 Text Recognition

Functions

- `template<typename L >`
`void scribo::text::recognition (line_set< L > &lines, const char *language)`
- `template<typename I >`
`void scribo::text::recognition (const Image< I > &line, const char *language, const std::string &output_file=std::string())`

6.21.1 Detailed Description

All routines/algorithms to recognize text.

6.21.2 Function Documentation

6.21.2.1 `template<typename L > void scribo::text::recognition (line_set< L > & lines, const char * language)`

Passes the text bboxes to Tesseract (OCR).

Parameters

<i>in</i>	<i>lines</i>	The lines of text.
<i>in</i>	<i>language</i>	The language which should be recognized by Tesseract. (fra, en, ...)

6.21.2.2 `template<typename I > void scribo::text::recognition (const Image< I > & line, const char * language, const std::string & output_file = std::string())`

Recognize text from an image.

Parameters

<i>in</i>	<i>line</i>	Image of text line.
<i>in</i>	<i>language</i>	The language which should be recognized by Tesseract. (fra, en, ...)
<i>in</i>	<i>output_file</i>	If set, store the recognized text in this file.

6.22 Text Extraction

Functions

- `template<typename I , typename N >`
`line_set< typename`
`mln::trait::ch_value< I,`
`scribo::def::lbl_type >::ret > scribo::text::extract_lines` (const **Image**< I > &input, const **Neighborhood**< N > &nbh, const typename `mln::trait::ch_value< I, bool >::ret` &separators)
- `template<typename I , typename N >`
`line_set< typename`
`mln::trait::ch_value< I,`
`scribo::def::lbl_type >::ret > scribo::text::extract_lines` (const **Image**< I > &input, const **Neighborhood**< N > &nbh)
- `template<typename L >`
`line_set< typename`
`mln::trait::ch_value< L,`
`scribo::def::lbl_type >::ret > scribo::text::extract_lines` (const component_set< L > &input)
- `template<typename I , typename J , typename N >`
`line_set< typename`
`mln::trait::ch_value< I,`
`scribo::def::lbl_type >::ret > scribo::text::extract_lines_with_features` (const **Image**< I > &input, const **Image**< J > &input_binary, const **Neighborhood**< N > &nbh, const typename `mln::trait::ch_value< I, bool >::ret` &separators)
- `template<typename I , typename J , typename N >`
`line_set< typename`
`mln::trait::ch_value< I,`
`scribo::def::lbl_type >::ret > scribo::text::extract_lines_with_features` (const **Image**< I > &input, const **Image**< J > &input_binary, const **Neighborhood**< N > &nbh)
- `template<typename I , typename N >`
`line_set< typename`
`mln::trait::ch_value< I,`
`scribo::def::lbl_type >::ret > scribo::text::extract_lines_wo_merge` (const **Image**< I > &input, const **Neighborhood**< N > &nbh, const typename `mln::trait::ch_value< I, bool >::ret` &separators)
- `template<typename I , typename N >`
`line_set< typename`
`mln::trait::ch_value< I,`
`scribo::def::lbl_type >::ret > scribo::text::extract_lines_wo_merge` (const **Image**< I > &input, const **Neighborhood**< N > &nbh)
- `template<typename L , typename N >`
`line_set< typename`
`mln::trait::ch_value< L,`
`scribo::def::lbl_type >::ret > scribo::text::extract_lines_wo_merge` (const document< L > &doc, const **Neighborhood**< N > &nbh)
- `template<typename L , typename N >`
`line_set< typename`
`mln::trait::ch_value< L,`
`scribo::def::lbl_type >::ret > scribo::text::extract_lines_wo_merge` (const document< L > &doc, const **Neighborhood**< N > &nbh, const typename `mln::trait::ch_value< L, bool >::ret` &separators)
- `template<typename L >`
`paragraph_set< L > scribo::text::extract_paragraphs` (line_set< L > &lines, const **image2d**< **bool** > &input)
- `template<typename L >`
`paragraph_set< L > scribo::text::extract_paragraphs_hdoc` (line_set< L > &lines, const **image2d**< **bool** > &input)
- `template<typename L >`
`line_links< L > scribo::text::link_lines` (const line_set< L > &lines)

- `template<typename L >`
`line_set< L > scribo::text::merging (const scribo::line_set< L > &lines)`
- `template<typename L >`
`line_set< L > scribo::text::merging_hdoc (const scribo::line_set< L > &lines)`

6.22.1 Detailed Description

All routines/algorithms to extract text.

6.22.2 Function Documentation

6.22.2.1 `template<typename I, typename N > line_set<typename mIn::trait::ch_value< I, scribo::def::lbl_type >::ret> scribo::text::extract_lines (const Image< I > & input, const Neighborhood< N > & nbh, const typename mIn::trait::ch_value< I, bool >::ret & separators)`

Extract lines of text in a binary image.

Parameters

<code>in</code>	<code>input</code>	A binary image.
<code>in</code>	<code>nbh</code>	A neighborhood used for labeling.
<code>in</code>	<code>separators</code>	A binary image with separator information.

Returns

A set of lines.

6.22.2.2 `template<typename I, typename N > line_set<typename mIn::trait::ch_value< I, scribo::def::lbl_type >::ret> scribo::text::extract_lines (const Image< I > & input, const Neighborhood< N > & nbh)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts.

6.22.2.3 `template<typename L > line_set<typename mIn::trait::ch_value< L, scribo::def::lbl_type >::ret> scribo::text::extract_lines (const component_set< L > & input)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts.

6.22.2.4 `template<typename I, typename J, typename N > line_set<typename mIn::trait::ch_value< I, scribo::def::lbl_type >::ret> scribo::text::extract_lines_with_features (const Image< I > & input, const Image< J > & input_binary, const Neighborhood< N > & nbh, const typename mIn::trait::ch_value< I, bool >::ret & separators)`

Extract lines of text in a binary image.

Parameters

<code>in</code>	<code>input</code>	Original color image.
<code>in</code>	<code>input_binary</code>	A binary image.
<code>in</code>	<code>nbh</code>	A neighborhood used for labeling.
<code>in</code>	<code>separators</code>	A binary image with separator information.

Returns

A set of lines.

```
6.22.2.5  template<typename I , typename J , typename N > line_set<typename mln::trait::ch_value< I ,
        scribo::def::lbl_type >::ret> scribo::text::extract_lines_with_features ( const Image< I > & input, const
        Image< J > & input_binary, const Neighborhood< N > & nbh )
```

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts.

```
6.22.2.6  template<typename I , typename N > line_set<typename mln::trait::ch_value< I , scribo::def::lbl_type
        >::ret> scribo::text::extract_lines_wo_merge ( const Image< I > & input, const Neighborhood< N > & nbh,
        const typename mln::trait::ch_value< I, bool >::ret & separators )
```

Extract lines of text in a binary image.

Parameters

in	<i>input</i>	A binary image.
in	<i>nbh</i>	A neighborhood used for labeling.
in	<i>separators</i>	A binary image with separator information.

Returns

A set of lines.

```
6.22.2.7  template<typename I , typename N > line_set<typename mln::trait::ch_value< I , scribo::def::lbl_type
        >::ret> scribo::text::extract_lines_wo_merge ( const Image< I > & input, const Neighborhood< N > & nbh )
```

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts.

```
6.22.2.8  template<typename L , typename N > line_set<typename mln::trait::ch_value< L , scribo::def::lbl_type
        >::ret> scribo::text::extract_lines_wo_merge ( const document< L > & doc, const Neighborhood< N > & nbh )
```

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts.

```
6.22.2.9  template<typename L , typename N > line_set<typename mln::trait::ch_value< L , scribo::def::lbl_type
        >::ret> scribo::text::extract_lines_wo_merge ( const document< L > & doc, const Neighborhood< N > & nbh,
        const typename mln::trait::ch_value< L, bool >::ret & separators )
```

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts.

```
6.22.2.10 template<typename L > paragraph_set<L> scribo::text::extract_paragraphs ( line_set< L > & lines, const
        image2d< bool > & input ) [inline]
```

Extract paragraphs.

6.22.2.11 `template<typename L> paragraph_set<L> scribo::text::extract_paragraphs_hdoc (line_set< L> & lines, const image2d< bool> & input)`

Extract paragraph in old documents.

6.22.2.12 `template<typename L> line_links<L> scribo::text::link_lines (const line_set< L> & lines)`

Link text lines to prepare paragraph construction.

Lines MUST be tagged as `line::Text` to be considered in this processing.

6.22.2.13 `template<typename L> line_set<L> scribo::text::merging (const scribo::line_set< L> & lines)`

Merge text component in order to reconstruct text lines.

Parameters

<code>in</code>	<code>lines</code>	A line set.
-----------------	--------------------	-------------

Returns

A new line set. Line ids are preserved and merged lines (not valid anymore) are tagged with `line::Merged`. The lines produced with this algorithm (valid lines) are tagged with `line::None`. Line type is also set either with `line::Text` or `line::Punctuation`.

6.22.2.14 `template<typename L> line_set<L> scribo::text::merging_hdoc (const scribo::line_set< L> & lines)`

Merge text component in order to reconstruct text lines in old documents.

Parameters

<code>in</code>	<code>lines</code>	A line set.
-----------------	--------------------	-------------

Returns

A new line set. Line ids are preserved and merged lines (not valid anymore) are tagged with `line::Merged`. The lines produced with this algorithm (valid lines) are tagged with `line::None`. Line type is also set either with `line::Text` or `line::Punctuation`.

6.23 Toolchains

Modules

- [Documents](#)
- [Pictures](#)

Functions

- `QSet< QString >` [scribo::toolchain::nepomuk::text_extraction](#) (const QImage &input, const QString &language)

6.23.1 Detailed Description

Full toolchains performing content analysis and extraction.

6.23.2 Function Documentation

6.23.2.1 `QSet<QString>` `scribo::toolchain::nepomuk::text_extraction` (const QImage & *input*, const QString & *language*)

Extract text from a document.

This is a convenient routine to be used in Nepomuk.

Parameters

<i>in</i>	<i>input</i>	A document image.
<i>in</i>	<i>language</i>	The main language used in the input document image. Improve text recognition quality if accurate.

Returns

A set of recognized words.

Don't forget to define NDEBUG for compilation to disable debug checks.

Depending on your version of Tesseract (OCR) you may define HAVE_TESSERACT_2 or HAVE_TESSERACT_3 .

6.24 Pictures

Modules

- [Preprocessing](#)
- [Processing](#)

6.24.1 Detailed Description

Full toolchains performing content analysis and extraction in pictures.

6.25 Documents

Modules

- [Preprocessing](#)
- [Processing](#)

6.25.1 Detailed Description

Full toolchains performing content analysis and extraction in document images.

6.26 Preprocessing

Classical preprocessing toolchains for pictures.

6.27 Preprocessing

Functions

- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::toolchain::text_in_doc_preprocess` (const **Image**< I > &input, **bool** enable_fg_bg, **unsigned** lambda, **double** K, **bool** enable_deskew, **bool** verbose)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::toolchain::text_in_doc_preprocess` (const **Image**< I > &input, **bool** enable_fg_bg, **bool** verbose)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::toolchain::text_in_doc_preprocess` (const **Image**< I > &input, **unsigned** lambda, **bool** verbose)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::toolchain::text_in_doc_preprocess` (const **Image**< I > &input, **unsigned** lambda, **double** k2, **double** k3, **double** k4, **bool** enable_fg_bg, **Image**< I > &fg, **bool** enable_deskew, **bool** verbose)
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret scribo::toolchain::text_in_doc_preprocess` (const **Image**< I > &input, **unsigned** lambda, **bool** enable_fg_bg, **Image**< I > &fg, **bool** enable_deskew, **bool** verbose)

6.27.1 Detailed Description

Classical preprocessing toolchains for document image.

6.27.2 Function Documentation

6.27.2.1 `template<typename I > mln::trait::ch_value< I, bool >::ret scribo::toolchain::text_in_doc_preprocess (const Image< I > & input, bool enable_fg_bg, unsigned lambda, double K, bool enable_deskew, bool verbose)`

Preprocess a document before looking for its content.

Parameters

<code>in</code>	<code>input</code>	An image.
<code>in</code>	<code>enable_fg_bg</code>	Enable/Disable background removal.
<code>in</code>	<code>lambda</code>	Parameter to the background removal. Maximum area of foreground objects.
<code>in</code>	<code>K</code>	Binanzation threshold parameter. Use the same value for all scales.(Default 0.34)
<code>in</code>	<code>enable_deskew</code>	Deskew document.
<code>in</code>	<code>verbose</code>	Enable/Disable debug output on std::cout.

If `enable_fg_bg` is set to 'True' then a background removal is performed. Its parameter `lambda` is automatically set according to the input image size.

6.27.2.2 `template<typename I > mln::trait::ch_value< I, bool >::ret scribo::toolchain::text_in_doc_preprocess (const Image< I > & input, bool enable_fg_bg, bool verbose)`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. K is set to 0.34.

6.27.2.3 `template<typename I> mIn::trait::ch_value< I, bool >::ret scribo::toolchain::text_in_doc_preprocess (const Image< I > & input, unsigned lambda, bool verbose)`

Preprocess a document before looking for its content.

Parameters

in	<i>input</i>	An image.
in	<i>lambda</i>	Parameter to the background removal.
in	<i>verbose</i>	Enable/Disable debug information printed on std::cout.

If lambda is set to '0' no background removal is performed. Otherwise, a background removal is performed with the given lambda value.

6.27.2.4 `template<typename I> mIn::trait::ch_value< I, bool >::ret scribo::toolchain::text_in_doc_preprocess (const Image< I > & input, unsigned lambda, double k2, double k3, double k4, bool enable_fg_bg, Image< I > & fg, bool enable_deskew, bool verbose)`

Preprocess a document before looking for its content. This methods relies on a multi-scale implementation of Sauvola's binarization.

Parameters

in	<i>input</i>	An image.
in	<i>lambda</i>	Parameter to the background removal.
in	<i>k2</i>	Binarization threshold parameter for scale 2. (Default 0.34)
in	<i>k3</i>	Binarization threshold parameter for scale 3. (Default 0.34)
in	<i>k4</i>	Binarization threshold parameter for scale 4. (Default 0.34)
in	<i>enable_fg_bg</i>	If set to True tries to identify background components and remove them.
in, out	<i>fg</i>	The foreground layer of input.
in	<i>enable_deskew</i>	Deskew document.
in	<i>verbose</i>	Enable/Disable debug information printed on std::cout.

If lambda is set to '0' no background removal is performed. Otherwise, a background removal is performed with the given lambda value.

6.27.2.5 `template<typename I> mIn::trait::ch_value< I, bool >::ret scribo::toolchain::text_in_doc_preprocess (const Image< I > & input, unsigned lambda, bool enable_fg_bg, Image< I > & fg, bool enable_deskew, bool verbose)`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

6.28 Processing

Functions

- `template<typename I >`
`component_set< typename`
`mln::trait::ch_value< I,`
`def::lbl_type >::ret > scribo::toolchain::text_in_picture` (const **Image**< I > &input_rgb_orig, **bool** bg_
removal, **bool** multi_scale_bin, **unsigned** max_dim_size=0, **unsigned** lambda=0, **bool** verbose=false)

6.28.1 Detailed Description

Classical processing toolchains for pictures.

6.28.2 Function Documentation

- 6.28.2.1 `template<typename I > component_set<typename mln::trait::ch_value< I , def::lbl_type >::ret>`
`scribo::toolchain::text_in_picture (const Image< I > & input_rgb_orig, bool bg_removal, bool multi_scale_bin,`
`unsigned max_dim_size = 0, unsigned lambda = 0, bool verbose = false)`

Localize text in pictures.

Parameters

<i>in</i>	<i>input_rgb_orig</i>	A RGB image.
<i>in</i>	<i>bg_removal</i>	If set to True tries to identify background components and remove them.
<i>in</i>	<i>multi_scale_bin</i>	Sauvola's binarization is used. Tells whether to use a multi-scale version or not.
<i>in</i>	<i>max_dim_size</i>	The maximum height or width allowed for the input image. If larger, it is resized.
<i>in</i>	<i>lambda</i>	Maximum area of components considered as foreground objects. If set to '0' (default), it is fixed automatically.
<i>in</i>	<i>verbose</i>	Enable/Disable debug output on std::cout.

Returns

A component set corresponding to the text components.

6.29 Processing

Functions

- `template<typename I, typename J >`
`document< typename`
`mln::trait::ch_value< I,`
`def::lbl_type >::ret > scribo::toolchain::content_in_doc (const Image< I > &input, const Image< J > &input_`
`_preproc, bool denoise, bool find_line_seps=true, bool find_whitespace_seps=true, bool enable_ocr=true,`
`const std::string &language=std::string("eng"), bool verbose=false)`
- `template<typename I, typename J >`
`document< typename`
`mln::trait::ch_value< I,`
`def::lbl_type >::ret > scribo::toolchain::content_in_hdoc (const Image< I > &input, const Image< J >`
`&input_preproc, bool denoise, bool find_line_seps=true, bool find_whitespace_seps=true, bool enable_`
`ocr=true, const std::string &language=std::string("eng"))`
- `template<typename I >`
`line_set< typename`
`mln::trait::ch_value< I,`
`def::lbl_type >::ret > scribo::toolchain::text_in_doc (const Image< I > &input, bool denoise, const std::`
`string &language=std::string("eng"), bool find_line_seps=true, bool find_whitespace_seps=true, bool ver-`
`bose=false)`

6.29.1 Detailed Description

Classical processing toolchains for document image.

6.29.2 Function Documentation

- 6.29.2.1 `template<typename I, typename J > document<typename mln::trait::ch_value< I, def::lbl_type >::ret>`
`scribo::toolchain::content_in_doc (const Image< I > &input, const Image< J > &input_preproc, bool denoise,`
`bool find_line_seps = true, bool find_whitespace_seps = true, bool enable_ocr = true, const std::string &`
`language = std::string("eng"), bool verbose = false)`

Analyse and extract document image content.

Parameters

<code>in</code>	<code>input</code>	A RGB image.
<code>in</code>	<code>input_preproc</code>	A Binary image.
<code>in</code>	<code>denoise</code>	Remove too small components (<= 2 pixels).
<code>in</code>	<code>find_line_seps</code>	Enable/Disable lookup for vertical/horizontal separators.
<code>in</code>	<code>find_whitespace_seps</code>	Enable/Disable lookup for vertical alignments and whitespaces. Helps finding text layout.
<code>in</code>	<code>enable_ocr</code>	Enable/Disable text recognition.
<code>in</code>	<code>language</code>	Main language used in the input document. (OCR settings)
<code>in</code>	<code>verbose</code>	Enable/Disable debug information printed on std::cout.

Returns

A document structure.

```
6.29.2.2 template<typename I, typename J> document<typename mIn::trait::ch_value< I, def::lbl_type >::ret>
scribo::toolchain::content_in_hdoc ( const Image< I> & input, const Image< J> & input_preproc, bool denoise,
bool find_line_seps = true, bool find_whitespace_seps = true, bool enable_ocr = true, const std::string &
language = std::string("eng") )
```

Analyse and extract content in a historical/degraded document.

Parameters

in	<i>input</i>	A RGB image.
in	<i>input_preproc</i>	A Binary image.
in	<i>denoise</i>	Remove too small components (≤ 2 pixels).
in	<i>find_line_seps</i>	Enable/Disable lookup for vertical/horizontal separators.
in	<i>find_whitespace_seps</i>	Enable/Disable lookup for vertical alignments and whitespaces. Helps finding text layout.
in	<i>enable_ocr</i>	Enable/Disable text recognition.
in	<i>language</i>	Main language used in the input document. (OCR settings)

Returns

A document structure.

```
6.29.2.3 template<typename I> line_set<typename mIn::trait::ch_value< I, def::lbl_type >::ret>
scribo::toolchain::text_in_doc ( const Image< I> & input, bool denoise, const std::string & language =
std::string("eng"), bool find_line_seps = true, bool find_whitespace_seps = true, bool verbose =
false )
```

Extract text lines from a document image.

Parameters

in	<i>input</i>	A RGB image.
in	<i>denoise</i>	Remove too small components (≤ 2 pixels).
in	<i>language</i>	Main language used in the input document. (OCR settings)
in	<i>find_line_seps</i>	Enable/Disable lookup for vertical/horizontal separators.
in	<i>find_whitespace_seps</i>	Enable/Disable lookup for vertical alignments and whitespaces. Helps finding text layout.
in	<i>verbose</i>	Enable/Disable debug information printed on std::cout.

Returns

A line set including text information and recognized text.

Chapter 7

Namespace Documentation

7.1 scribo Namespace Reference

Namespaces

- namespace [binarization](#)
- namespace [component](#)
- namespace [core](#)
- namespace [debug](#)
- namespace [draw](#)
- namespace [filter](#)
- namespace [make](#)
- namespace [postprocessing](#)
- namespace [preprocessing](#)
- namespace [primitive](#)
- namespace [table](#)
- namespace [text](#)

Classes

- struct [component_features_data](#)
- class [component_info](#)
Component information data structure.
- class [component_set](#)
Represents all the components in a document image.
- class [DMax_Functor](#)
Dmax functor concept.
- class [doc_serializer](#)
Link functor concept.
- class [document](#)
Represent document data and structure.
- class [group_info](#)
- class [line_info](#)
- class [line_links](#)
Line links representation.
- class [line_set](#)
Lines container.
- class [Link_Functor](#)

- *Link functor concept.*
- class [object_groups](#)
Object group representation.
- class [object_links](#)
Object links representation.
- class [paragraph_info](#)
Paragraph structure information.
- class [paragraph_set](#)
Paragraph container.
- class [Serializable](#)
Concept for objects that can be serialized.
- class [SerializeVisitor](#)
Link functor concept.

Typedefs

- typedef [mln::util::object_id](#)
< scribo::ComponentId,
unsigned > [component_id_t](#)
- typedef [mln::util::object_id](#)
< scribo::LineId, **unsigned** > [line_id_t](#)
- typedef [mln::util::object_id](#)
< scribo::ParagraphId,
unsigned > [paragraph_id_t](#)

Functions

- template<typename P >
mln::util::couple< P, P > [central_sites](#) (const **box**< P > &b, **unsigned** dim)
- template<typename I, typename L >
void [erase_objects](#) (**Image**< I > &input, const [component_set](#)< L > &comps)
- template<typename I, typename F >
mln::trait::ch_value< I,
double >::ret [init_integral_image](#) (const **Image**< I > &input_, F &func)
- std::ostream & **operator**<< (std::ostream &ostr, const [component_features_data](#) &data)
- template<typename L >
std::ostream & **operator**<< (std::ostream &ostr, const [paragraph_info](#)< L > &info)
- template<typename L >
std::ostream & **operator**<< (std::ostream &ostr, const [line_links](#)< L > &links)
- template<typename L >
std::ostream & **operator**<< (std::ostream &ostr, const [component_info](#)< L > &info)
- template<typename L >
std::ostream & **operator**<< (std::ostream &ostr, const [object_groups](#)< L > &groups)
- template<typename L >
std::ostream & **operator**<< (std::ostream &ostr, const [line_set](#)< L > &lines)
- template<typename L >
std::ostream & **operator**<< (std::ostream &ostr, const [line_info](#)< L > &info)
- **bool operator==** (const [component_features_data](#) &lhs, const [component_features_data](#) &rhs)
- **bool operator==** (const [group_info](#) &lhs, const [group_info](#) &rhs)
- template<typename L >
bool operator== (const [paragraph_set](#)< L > &lhs, const [paragraph_set](#)< L > &rhs)
- template<typename L >
bool operator== (const [paragraph_info](#)< L > &lhs, const [paragraph_info](#)< L > &rhs)

- `template<typename L >`
bool operator== (const [line_links](#)< L > &lhs, const [line_links](#)< L > &rhs)
- `template<typename L >`
bool operator== (const [component_info](#)< L > &lhs, const [component_info](#)< L > &rhs)
- `template<typename L >`
bool operator== (const [object_groups](#)< L > &lhs, const [object_groups](#)< L > &rhs)
- `template<typename L >`
bool operator== (const [document](#)< L > &lhs, const [document](#)< L > &rhs)
- `template<typename L >`
bool operator== (const [line_set](#)< L > &lhs, const [line_set](#)< L > &rhs)
- `template<typename L >`
bool operator== (const [line_info](#)< L > &lhs, const [line_info](#)< L > &rhs)

7.1.1 Detailed Description

The main namespace of the Scribo module. try to determine the type of a component.

FIXME: provide a version for binary images.

FIXME: share code with [filter/object_groups_with_holes.hh](#) FIXME: Merge the two following routines.

Function increasing values to highlight areas.

FIXME: return type too restrictive!

FIXME: result is not consistent with other binarization methods.

Namespace of the whole project.

The result is inverted because of the threshold functor: should we invert the test in this functor?

FIXME: Use a size ratio in both overloads.

7.1.2 Typedef Documentation

7.1.2.1 `typedef mln::util::object_id<scribo::ComponentId, unsigned> scribo::component_id_t`

The type of the component ids.

This id is mainly used in structures like [scribo::component_set](#) and [scribo::component_info](#) structures. It refers to the actual label used in the underlying labeled component image stored in [scribo::component_set](#).

Definition at line 48 of file [component_info.hh](#).

7.1.3 Function Documentation

7.1.3.1 `template<typename P > mln::util::couple<P,P> scribo::central_sites (const box< P > & b, unsigned dim)`

Returns the edge central sites of a box.

Parameters

<i>in</i>	<i>b</i>	the bbbox
<i>in</i>	<i>dim</i>	the dimension used to compute the site.

If `dim == 0`, returns the left and right central sites.

```
|-----|
X X
|-----|
```

If `dim == 1`, returns the top and bottom central sites.

```
|—X—|
|
|
|—X—|
```

... And so on.

7.1.3.2 `template<typename I, typename L> void scribo::erase_objects (Image< I > &input, const component_set< L > &comps)`

Remove labeled components from a binary image.

Parameters

<code>in, out</code>	<code>input</code>	A binary image.
<code>in</code>	<code>comps</code>	A set of components to be erased.

7.2 scribo::binarization Namespace Reference

Functions

- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret global_threshold (const Image< I > &input, const typename I::value &threshold)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret global_threshold_auto (const Image< I > &input)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret kim (const Image< I > &input, unsigned window_size, double k)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret kim (const Image< I > &input, unsigned window_size)`
- `template<typename I, typename T>`
`mln::trait::ch_value< I, bool >`
`::ret local_threshold (const Image< I > &input, const Image< T > &threshold)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret niblack (const Image< I > &input, unsigned window_size, double K)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret niblack (const Image< I > &input, unsigned window_size)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret niblack (const Image< I > &input)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret niblack_fast (const Image< I > &input, unsigned window_size, double K)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret niblack_fast (const Image< I > &input, unsigned window_size)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret niblack_fast (const Image< I > &input)`

- `template<typename I, typename J>`
`mln::trait::ch_value< I,`
`value::int_u8 >::ret niblack_threshold (const Image< I > &input, unsigned window_size, double K)`
- `template<typename I>`
`mln::trait::ch_value< I,`
`value::int_u8 >::ret niblack_threshold (const Image< I > &input, unsigned window_size)`
- `template<typename I>`
`mln::trait::ch_value< I,`
`value::int_u8 >::ret niblack_threshold (const Image< I > &input)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret otsu (const Image< I > &input)`
- `template<typename I>`
`I::value otsu_threshold (const Image< I > &input)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret sauvola (const Image< I > &input, unsigned window_size, double K)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret sauvola (const Image< I > &input, unsigned window_size)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret sauvola (const Image< I > &input)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret sauvola_ms (const Image< I > &input_1, unsigned w_1, unsigned s, image2d< mln::util::couple<`
`double, double > > &integral_sum_sum_2)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret sauvola_ms (const Image< I > &input_1, unsigned w_1, unsigned s)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret sauvola_ms (const Image< I > &input_1, unsigned w_1)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret sauvola_ms (const Image< I > &input_1, unsigned w_1, unsigned s, double k2, double k3, double`
`k4)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret sauvola_ms (const Image< I > &input_1, unsigned w_1, unsigned s, double all_k)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret sauvola_ms_split (const Image< I > &input_1, unsigned w_1, unsigned s, unsigned min_ntrue, dou-`
`ble k2, double k3, double k4)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret sauvola_ms_split (const Image< I > &input_1, unsigned w_1, unsigned s, unsigned min_ntrue, dou-`
`ble K)`
- `template<typename I>`
`mln::trait::ch_value< I, bool >`
`::ret sauvola_ms_split (const Image< I > &input_1, unsigned w_1, unsigned s, unsigned min_ntrue)`
- `template<typename I, typename J>`
`mln::trait::ch_value< I,`
`value::int_u8 >::ret sauvola_threshold (const Image< I > &input, unsigned window_size, double K)`
- `template<typename I>`
`mln::trait::ch_value< I,`
`value::int_u8 >::ret sauvola_threshold (const Image< I > &input, unsigned window_size)`

- `template<typename I >`
`mln::trait::ch_value< I,`
`value::int_u8 >::ret sauvola_threshold (const Image< I > &input)`
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret singh (const Image< I > &input, unsigned window_size, double K)`
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret singh (const Image< I > &input, unsigned window_size)`
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret singh (const Image< I > &input)`
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret wolf (const Image< I > &input, unsigned window_size, double K)`
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret wolf (const Image< I > &input, unsigned window_size)`
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret wolf (const Image< I > &input)`
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret wolf_fast (const Image< I > &input, unsigned window_size, double K)`
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret wolf_fast (const Image< I > &input, unsigned window_size)`
- `template<typename I >`
`mln::trait::ch_value< I, bool >`
`::ret wolf_fast (const Image< I > &input)`

7.2.1 Detailed Description

Namespace of binarization routines.

7.2.2 Function Documentation

7.2.2.1 `template<typename I > mln::trait::ch_value< I, bool >::ret scribo::binarization::global_threshold (const Image< I > & input, const typename I::value & threshold)`

Binarize an image using a global threshold value.

For a site 'p' in `input` image:

`output(p) = input(p) >= threshold`

Parameters

<code>in</code>	<code>input</code>	A grayscale image.
<code>in</code>	<code>threshold</code>	A value.

Returns

A boolean image.

7.2.2.2 `template<typename I> mIn::trait::ch_value< I, bool >::ret scribo::binarization::global_threshold_auto (const Image< I > & input)`

Simple binarization of a gray-level document.

Automatically find a global threshold for the given image.

Parameters

<i>in</i>	<i>input</i>	A gray-level image.
-----------	--------------	---------------------

Returns

A Boolean image.

7.2.2.3 `template<typename I, typename T> mIn::trait::ch_value< I, bool >::ret scribo::binarization::local_threshold (const Image< I > & input, const Image< T > & threshold)`

Binarize an image using a threshold image.

The threshold image is used to specify a specific threshold for each site.

For a site 'p' in *input* image:

$output(p) = input(p) \geq threshold(p)$

Precondition

$input.domain() == threshold.domain()$

Parameters

<i>in</i>	<i>input</i>	A grayscale image.
<i>in</i>	<i>threshold</i>	A grayscale image.

Returns

A boolean image.

7.2.2.4 `template<typename I> mIn::trait::ch_value< I, bool >::ret scribo::binarization::sauvola_ms (const Image< I > & input_1, unsigned w_1, unsigned s, double k2, double k3, double k4)`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. Allow to specify a different k parameter for each scale.

7.2.2.5 `template<typename I> mIn::trait::ch_value< I, bool >::ret scribo::binarization::sauvola_ms (const Image< I > & input_1, unsigned w_1, unsigned s, double all_k)`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. Allow to specify the same k parameter for all scales.

7.2.2.6 `template<typename I> mIn::trait::ch_value< I, bool >::ret scribo::binarization::singh (const Image< I > & input, unsigned window_size, double K)`

Convert an image into a binary image.

Parameters

in	<i>input</i>	An image.
in	<i>window_size</i>	The window size.
in	<i>K</i>	Singh's formulae constant.

Returns

A binary image.

7.2.2.7 `template<typename I> mln::trait::ch_value< I, bool >::ret scribo::binarization::singh (const Image< I > & input, unsigned window_size)`

Convert an image into a binary image.

Singh's formulae constant K is set to 0.34.

Parameters

in	<i>input</i>	An image.
in	<i>window_size</i>	The window size.

Returns

A binary image.

7.2.2.8 `template<typename I> mln::trait::ch_value< I, bool >::ret scribo::binarization::singh (const Image< I > & input)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. The window size is set to 11.

7.3 scribo::component Namespace Reference

Enumerations

- enum [Tag](#) { [None](#) = 0, [Ignored](#) }
- enum [Type](#) { [Undefined](#) = 0, [Character](#), [VerticalLineSeparator](#), [HorizontalLineSeparator](#), [WhitespaceSeparator](#), [Noise](#), [Punctuation](#), [Image](#), [DropCapital](#) }

Functions

- `std::ostream & operator<< (std::ostream &ostr, const Tag &tag)`
- `std::ostream & operator<< (std::ostream &ostr, const Type &type)`
- `Tag str2tag (const std::string &str)`
- `Type str2type (const std::string &str)`

7.3.1 Detailed Description

Namespace of classes related to components.

7.3.2 Enumeration Type Documentation

7.3.2.1 enum scribo::component::Tag

All possible tags that can be used for components.

[component::Tag](#) values can be converted to String using [operator<<\(std::ostream &ostr, const Tag &tag\)](#). String values can be converted to [component::Tag](#) using [str2tag\(\)](#).

See Also

[scribo::component_set](#) [scribo::component_info](#)

Enumerator:

None No tag set.

Ignored To be ignored in further processing.

Definition at line 54 of file component.hh.

7.3.2.2 enum scribo::component::Type

All possible types of components.

[component::Type](#) values can be converted to String using [operator<<\(std::ostream &ostr, const Type &type\)](#). String values can be converted to [component::Type](#) using [str2type\(\)](#).

See Also

[scribo::component_set](#) [scribo::component_info](#) [str2type\(\)](#)

Enumerator:

Undefined No type defined.

Character Character.

VerticalLineSeparator Vertical line separator.

HorizontalLineSeparator Horizontal line separator.

WhitespaceSeparator whitespace separator.

Noise Noise.

Punctuation Punctuation.

Image Image.

DropCapital Drop capital.

Definition at line 69 of file component.hh.

7.3.3 Function Documentation

7.3.3.1 std::ostream& scribo::component::operator<< (std::ostream & ostr, const Tag & tag)

Operator allowing [scribo::component::Tag](#) to be printed out.

7.3.3.2 std::ostream& scribo::component::operator<< (std::ostream & ostr, const Type & type)

Operator allowing [scribo::component::Type](#) to be printed out.

7.3.3.3 Tag scribo::component::str2tag (const std::string & str)

Convert a std::string to a [scribo::component::Tag](#).

7.3.3.4 Type scribo::component::str2type (const std::string & str)

Convert a std::string to a [scribo::component::Type](#).

7.4 scribo::core Namespace Reference

7.4.1 Detailed Description

Namespace of core routines.

7.5 scribo::debug Namespace Reference

Classes

- struct [arg_data](#)
- struct [opt_data](#)
- class [option_parser](#)
- struct [toggle_data](#)

Enumerations

- enum [Level](#) {
None = 0, **Special**, **Results**, **AuxiliaryResults**,
All, **InvalidLevel** }
- enum [VerboseMode](#) {
Mute = 0, **UserDebug**, **Time**, **Low**,
Medium, **Full**, **Invalid** }

Functions

- template<typename I, typename L >
mln::trait::ch_value< I,
value::rgb8 >::ret [alignment_decision_image](#) (const **Image**< I > &input, const [object_links](#)< L > &links,
const [object_links](#)< L > &filtered_links, const anchor::Type &anchor)
- template<typename I, typename L >
mln::trait::ch_value< I,
value::rgb8 >::ret [bboxes_enlarged_image](#) (const **Image**< I > &input, const [line_set](#)< L > &lines, const
value::rgb8 &text_value, const **value::rgb8** &non_text_value)
- template<typename I, typename L >
mln::trait::ch_value< I,
value::rgb8 >::ret [bboxes_enlarged_image](#) (const **Image**< I > &input, const [line_set](#)< L > &lines)
- template<typename I >
mln::trait::ch_value< I,
value::rgb8 >::ret [bboxes_image](#) (const **Image**< I > &input, const **mln::util::array**< **box**< typename I::site
> > &bboxes, const **value::rgb8** &value)

- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret bboxes_image (const Image< I > &input, const line_set< L > &lines, const value::rgb8`
`&value)`
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret bboxes_image (const Image< I > &input, const line_set< L > &lines)`
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret bboxes_image (const Image< I > &input, const component_set< L > &comps, const`
`value::rgb8 &value)`
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret char_space_image (const Image< I > &input, const line_set< L > &line, const value-`
`::rgb8 &v=literal::cyan)`
- `bool check_ocr_lang (const std::vector< const char * > &args)`
- `bool check_sauvola_first_subsampling (const std::vector< const char * > &args)`
- `bool check_sauvola_split_ntrue (const std::vector< const char * > &args)`
- `bool check_verbose_mode (const std::vector< const char * > &args)`
- `bool check_xml_format (const std::vector< const char * > &args)`
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret decision_image (const Image< I > &input, const object_groups< L > &groups, const`
`object_groups< L > &filtered_groups, anchor::Type anchor)`
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret decision_image (const Image< I > &input, const object_links< L > &links, const object-`
`_links< L > &filtered_links, anchor::Type anchor)`
- `template<typename I>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret highlight_text_area (const Image< I > &input, const mln::util::array< box< typename`
`I::site > > &bbox)`
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret highlight_text_area (const Image< I > &input, const line_set< L > &lines)`
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret highlight_text_area (const Image< I > &input, const scribo::component_set< L > &com-`
`ponents)`
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret line_info_image (const Image< I > &input, const line_set< L > &line)`
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret linked_bboxes_image (const Image< I > &input, const object_links< L > &array, const`
`value::rgb8 &box_value, const value::rgb8 &link_value, anchor::Type anchor)`
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret linked_bboxes_image (const Image< I > &input, const object_links< L > &array, const`
`value::rgb8 &box_value, const value::rgb8 &link_value)`
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret linked_bboxes_image (const Image< I > &input, const object_links< L > &left_link, const`
`object_links< L > &right_link, const value::rgb8 &box_value, const value::rgb8 &link_value, anchor::Type`
`anchor)`

- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret linked_bboxes_image` (const `Image< I >` &input, const `object_links< L >` &left_link,
const `object_links< L >` &right_link, const `value::rgb8` &box_value, const `value::rgb8` &left_link_value, const
`value::rgb8` &right_link_value, const `value::rgb8` &validated_link_value, anchor::Type anchor)
- `template<typename I, typename L, typename G>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret linked_bboxes_image` (const `Image< I >` &input, const `Graph< G >` &g, const `value-`
`::rgb8` &box_value, const `value::rgb8` &link_value, anchor::Type anchor)
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret links_decision_image` (const `Image< I >` &input_, const `object_links< L >` &links, const
`object_links< L >` &filtered_links)
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret links_image` (const `Image< I >` &input_, const `object_links< L >` &links, anchor::Type
anchor, `bool` draw_bboxes=true)
- `scribo::debug::internal::logger_ & logger ()`
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret looks_like_a_text_line_image` (const `Image< I >` &input, const `line_set< L >` &lines,
const `value::rgb8` &text_value, const `value::rgb8` &non_text_value)
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret looks_like_a_text_line_image` (const `Image< I >` &input, const `line_set< L >` &lines)
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret mean_and_base_lines_image` (const `Image< I >` &input, const `line_set< L >` &lines,
const `value::rgb8` &bbox_value, const `value::rgb8` &meanline_value, const `value::rgb8` &baseline_value)
- `template<typename I, typename L>`
`mln::trait::ch_value< I,`
`value::rgb8 >::ret mean_and_base_lines_image` (const `Image< I >` &input, const `line_set< L >` &lines)
- `template<typename L, typename L2>`
`void save_comp_diff` (const `component_set< L >` &comps_ref, const `component_set< L2 >` &comps_new,
const std::string &filename)
- `template<typename I>`
`void save_label_image` (const `Image< I >` &lbl, const typename `I::value` &nlabels, const char *filename)
- `template<typename I>`
`void save_table_image` (const `Image< I >` &input_, `mln::util::couple< mln::util::array< box< typename`
`I::site > >, mln::util::array< box< typename I::site > > >` tableboxes, const `value::rgb8` &bbox_color,
const std::string &filename)
- `template<typename S>`
`void save_table_image` (const `Site_Set< S >` &input_domain, `mln::util::couple< mln::util::array< box<`
`typename S::site > >, mln::util::array< box< typename S::site > > >` tableboxes, const `value::rgb8`
&bg_color, const `value::rgb8` &bbox_color, const std::string &filename)
- `template<typename I, typename L>`
`mln::trait::concrete< I >::ret text_areas_image` (const `Image< I >` &input_rgb, const `scribo::component_-`
`set< L >` &comps)
- `template<typename L>`
`image2d< value::rgb8 > text_color_image` (const `document< L >` &doc)
- `VerboseMode txt_to_verbose_mode` (const std::string &name)
- `int usage` (char *argv[], const char *desc, const char *args, const char *args_desc[][2])

7.5.1 Detailed Description

Namespace of debug routines.

7.5.2 Enumeration Type Documentation

7.5.2.1 enum scribo::debug::Level

Enum defining different level of image logging.

According to the debug level set in scribo::debug::logger_ the image may be saved or not.

Definition at line 57 of file logger.hh.

7.5.2.2 enum scribo::debug::VerboseMode

Enum defining different modes of text logging.

According to the verbose mode set in scribo::debug::logger_ the text may be logged or not.

Definition at line 72 of file logger.hh.

7.5.3 Function Documentation

7.5.3.1 VerboseMode scribo::debug::txt_to_verbose_mode (const std::string & name)

returns the corresponding verboseMode from its name.

7.5.3.2 int scribo::debug::usage (char * argv[], const char * desc, const char * args, const char * args_desc[][2])

Format a standard usage output.

Parameters

in	<i>argv</i>	Arguments passed to the program.
in	<i>desc</i>	Description of the program.
in	<i>args</i>	The expected arguments.
in	<i>args_desc</i>	The description of the expected arguments.

Returns

Return 1.

7.6 scribo::draw Namespace Reference

Functions

- template<typename I, typename L >
void [bounding_box_links](#) (Image< I > &input, const [object_links](#)< L > &link, const typename I::value &value, anchor::Type anchor)
- template<typename I, typename L >
void [bounding_box_links](#) (Image< I > &input, const [object_links](#)< L > &link, const typename I::value &value)
- template<typename I, typename L >
void [bounding_box_links](#) (Image< I > &input, const [object_links](#)< L > &left_link, const [object_links](#)< L > &right_link, const typename I::value &left_link_value, const typename I::value &right_link_value, const typename I::value &validated_link_value, anchor::Type anchor)
- template<typename I, typename G >
void [bounding_box_links](#) (Image< I > &input, const [Graph](#)< G > &g, const typename I::value &link_value)

- `template<typename I >`
`void bounding_boxes (Image< I > &input_, const mIn::util::array< box< typename I::site > > &boxes, const typename I::value &value)`
- `template<typename I , typename L >`
`void bounding_boxes (Image< I > &input_, const component_set< L > &components, const typename I::value &value)`
- `template<typename I , typename L >`
`void groups_bboxes (Image< I > &input_, const object_groups< L > &groups, const typename I::value &value)`
- `template<typename L , typename I >`
`void line_components (Image< I > &input_, const line_set< L > &lines, const line_info< L > &line, const typename I::value &value)`

7.6.1 Detailed Description

Namespace of drawing routines.

7.6.2 Function Documentation

7.6.2.1 `template<typename I , typename L > void scribo::draw::bounding_box_links (Image< I > &input, const object_links< L > &link, const typename I::value &value, anchor::Type anchor)`

Draw a list of bounding box links.

Parameters

in, out	<i>input</i>	An image where to draw.
in	<i>link</i>	component links.
in	<i>value</i>	Value used to draw links.
in	<i>anchor</i>	Anchor from where the links are drawn.

7.6.2.2 `template<typename I , typename L > void scribo::draw::bounding_box_links (Image< I > &input, const object_links< L > &link, const typename I::value &value)`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. The default anchor type is set to `anchor::Center`.

7.6.2.3 `template<typename I , typename L > void scribo::draw::bounding_box_links (Image< I > &input, const object_links< L > &left_link, const object_links< L > &right_link, const typename I::value &left_link_value, const typename I::value &right_link_value, const typename I::value &validated_link_value, anchor::Type anchor)`
`[inline]`

Draw left, right and validated lists of bounding box links.

Parameters

in, out	<i>input</i>	An image where to draw.
in	<i>left_link</i>	Component's left links.
in	<i>right_link</i>	Component's right links.
in	<i>left_link_value</i>	Value used to draw left links.
in	<i>right_link_value</i>	Value used to draw right links.
in	<i>validated_link_value</i>	Value used to draw validated links.
in	<i>anchor</i>	Anchor from where the links are drawn.

7.6.2.4 `template<typename I , typename G > void scribo::draw::bounding_box_links (Image< I > & input, const Graph< G > & g, const typename I::value & link_value) [inline]`

Draw a graph of bounding box links.

Draw from bounding box centers.

Parameters

<i>in, out</i>	<i>input</i>	An image where to draw.
<i>in</i>	<i>g</i>	The link graph.
<i>in</i>	<i>link_value</i>	The value used to draw the links.

7.6.2.5 `template<typename I > void scribo::draw::bounding_boxes (Image< I > & input_, const mln::util::array< box< typename I::site > > & boxes, const typename I::value & value)`

Draw a list of bounding boxes.

7.6.2.6 `template<typename I , typename L > void scribo::draw::bounding_boxes (Image< I > & input_, const component_set< L > & components, const typename I::value & value)`

Draw object bounding boxes.

7.6.2.7 `template<typename I , typename L > void scribo::draw::groups_bboxes (Image< I > & input_, const object_groups< L > & groups, const typename I::value & value) [inline]`

Draw a list of bounding boxes.

7.7 scribo::filter Namespace Reference

Functions

- `template<typename I , typename N , typename V > mln::trait::concrete< I >::ret_components_large (const Image< I > &input_, const Neighborhood< N > &nbh_, const V &label_type, unsigned max_size)`
- `template<typename L > component_set< L > components_large (const component_set< L > &components, unsigned max_size)`
- `template<typename L > component_set< L > components_on_border (const component_set< L > &components)`
- `template<typename I , typename N , typename V > mln::trait::concrete< I >::ret_components_small (const Image< I > &input, const Neighborhood< N > &nbh, V &nlabels, unsigned min_size)`
- `template<typename L > component_set< L > components_small (const component_set< L > &components, unsigned min_size)`
- `template<typename I , typename N , typename V > mln::trait::concrete< I >::ret_components_thin (const Image< I > &input_, const Neighborhood< N > &nbh_, const V &label_type, unsigned min_thickness)`
- `template<typename L > component_set< L > components_thin (const component_set< L > &components, unsigned min_thickness)`
- `template<typename L > component_set< L > components_v_thin (const component_set< L > &comps, unsigned min_thinness)`

- `template<typename L >`
`component_set< L > components_with_two_holes` (const `component_set< L >` &components, **unsigned** min_size)
- `template<typename P , typename V , typename G , typename F , typename FP >`
edge_image< void, **bool**, G > **graph_edges** (const **vertex_image**< P, V, G > &v_ima, const **Function**< F > &, const **Function**< FP > &)
- `template<typename L >`
void **images_in_paragraph** (document< L > &doc)
- `template<typename L >`
line_links< L > **line_links_x_height** (const **line_links**< L > &links)
- `template<typename L >`
object_groups< L > **object_groups_mean_width** (const **object_groups**< L > &groups, **float** width)
- `template<typename L >`
object_groups< L > **object_groups_size_ratio** (const **object_groups**< L > &groups, **float** max_size_ratio, **float** max_invalid_ratio_per_group)
- `template<typename L >`
object_groups< L > **object_groups_small** (const **object_groups**< L > &groups, **unsigned** n_links)
- `template<typename L >`
object_groups< L > **object_groups_with_holes** (const **object_groups**< L > &components, **unsigned** min_size)
- `template<typename L >`
object_links< L > **object_links_aligned** (const **object_links**< L > &links, **float** max_alpha, anchor::Type anchor)
- `template<typename L >`
object_links< L > **object_links_bbox_h_ratio** (const **object_links**< L > &links, **float** max_h_ratio)
- `template<typename L >`
object_links< L > **object_links_bbox_overlap** (const **object_links**< L > &links, **float** max_overlap_ratio)
- `template<typename L >`
object_links< L > **object_links_bbox_ratio** (const **object_links**< L > &links, **unsigned** dim, **float** max_ratio)
- `template<typename L >`
object_links< L > **object_links_bbox_w_ratio** (const **object_links**< L > &links, **float** max_w_ratio)
- `template<typename L >`
object_links< L > **object_links_bottom_aligned** (const **object_links**< L > &links, **float** max_alpha)
- `template<typename L >`
object_links< L > **object_links_center_aligned** (const **object_links**< L > &links, **float** max_alpha)
- `template<typename L >`
object_links< L > **object_links_left_aligned** (const **object_links**< L > &links, **float** max_alpha)
- `template<typename L >`
object_links< L > **object_links_non_aligned_simple** (const **object_links**< L > &links, anchor::Type anchor, **float** max_alpha)
- `template<typename L >`
object_links< L > **object_links_right_aligned** (const **object_links**< L > &links, **float** max_alpha)
- `template<typename L >`
object_links< L > **object_links_top_aligned** (const **object_links**< L > &links, **float** max_alpha)
- `template<typename I , typename N , typename V >`
mln::trait::concrete< I >::ret **objects_h_thick** (const **Image**< I > &input_, const **Neighborhood**< N > &nbh_, const V &label_type, **unsigned** max_thickness)
- `template<typename L >`
component_set< L > **objects_h_thick** (const **component_set**< L > &comps, **unsigned** max_thickness)
- `template<typename I , typename N , typename V >`
mln::trait::concrete< I >::ret **objects_h_thin** (const **Image**< I > &input_, const **Neighborhood**< N > &nbh_, const V &label_type, **unsigned** min_thinness)
- `template<typename L >`
component_set< L > **objects_h_thin** (const **component_set**< L > &comps, **unsigned** min_thinness)
- `template<typename L >`
void **objects_in_borders** (**component_set**< L > &components, **float** vratio, **float** hratio)

- `template<typename L >`
`component_set< L > objects_size_ratio` (const `component_set< L >` &comps, `float` min_size_ratio)
- `template<typename I , typename N , typename V >`
`mln::trait::concrete< I >::ret objects_thick` (const `Image< I >` &input_, const `Neighborhood< N >` &nbh_, const `V` &label_type, `unsigned` max_thickness)
- `template<typename L >`
`component_set< L > objects_thick` (const `component_set< L >` &components, `unsigned` max_thickness)
- `template<typename I , typename N , typename V >`
`mln::trait::concrete< I >::ret objects_v_thick` (const `Image< I >` &input_, const `Neighborhood< N >` &nbh_, const `V` &label_type, `unsigned` max_thickness)
- `template<typename L >`
`component_set< L > objects_v_thin` (const `component_set< L >` &comps, `unsigned` max_thickness)
- `template<typename I , typename N , typename V >`
`mln::trait::concrete< I >::ret objects_v_thin` (const `Image< I >` &input, const `Neighborhood< N >` &nbh, const `V` &label_type, `unsigned` min_thinness)
- `template<typename L >`
`component_set< L > objects_with_holes` (const `component_set< L >` &components, `unsigned` min_holes_count, `unsigned` min_size)
- `template<typename L >`
`paragraph_set< L > paragraphs_bbox_overlap` (const `paragraph_set< L >` &parset)
- `template<typename L >`
void `paragraphs_in_borders` (`document< L >` &doc)
- `template<typename L >`
void `paragraphs_in_image` (`document< L >` &doc)
- `template<typename L >`
void `separators_in_borders` (`document< L >` &doc, `float` vratio, `float` hratio)
- `template<typename L >`
void `separators_in_element` (`document< L >` &doc)
- `template<typename L >`
void `separators_in_paragraph` (`document< L >` &doc, `unsigned` hmin_size, `unsigned` vmin_size)
- `template<typename L >`
void `separators_vert_in_borders` (`document< L >` &doc)

7.7.1 Detailed Description

Namespace of filtering routines.

7.8 scribo::make Namespace Reference

Functions

- `std::string debug_filename` (const `std::string` &name)
- `template<typename I , typename N , typename V >`
`mln::util::graph influence_zone_graph` (const `Image< I >` &input_, const `Neighborhood< N >` &nbh_, const `V` &label_type, `unsigned` iz_dmax)
- `template<typename L >`
`scribo::line_set< L > line_set` (const `object_groups< L >` &groups)
- `template<typename L >`
`scribo::paragraph_set< L > paragraph` (const `line_links< L >` &llinks, const `line_links< L >` &rlinks)
- `template<typename L >`
`scribo::paragraph_set< L > paragraph` (const `scribo::line_set< L >` &lines)
- `template<typename L >`
`scribo::paragraph_set< L > paragraph` (const `line_links< L >` &llinks)

- `template<typename L >`
`mln::trait::ch_value< L, bool >`
`::ret text_blocks_image (const document< L > &doc, unsigned min_nlines)`
- `template<typename L >`
`mln::trait::ch_value< L, bool >`
`::ret text_components_image (const document< L > &doc)`

7.8.1 Detailed Description

Namespace of routines constructing objects.

7.8.2 Function Documentation

7.8.2.1 `std::string scribo::make::debug_filename (const std::string & name)`

Construct and returns a formatted output file name:

```
`input_filename`_`id`_`name`
```

See Also

`scribo::make::internal::debug_filename_prefix`

7.8.2.2 `template<typename I, typename N, typename V > mln::util::graph scribo::make::influence_zone_graph (const Image< I > & input_, const Neighborhood< N > & nbh_, const V & label_type, unsigned iz_dmax)`

Compute a labeled image of input, then compute an influence zone image and make a graph from it.

Parameters

<code>in</code>	<code>input_</code>	a binary image.
<code>in</code>	<code>nbh_</code>	a neighborhood.
<code>in</code>	<code>label_type</code>	The type of this argument is used as label type while labeling the image.
<code>in</code>	<code>iz_dmax</code>	Max distance of the influence zone.

Returns

a region adjacency graph.

7.8.2.3 `template<typename L > scribo::paragraph_set<L> scribo::make::paragraph (const line_links< L > & llinks, const line_links< L > & rlinks)`

Construct a paragraph set from line links information.

7.8.2.4 `template<typename L > scribo::paragraph_set<L> scribo::make::paragraph (const scribo::line_set< L > & lines)`

Construct a paragraph set from line set information.

7.8.2.5 `template<typename L > scribo::paragraph_set<L> scribo::make::paragraph (const line_links< L > & llinks)`

Construct a paragraph set from line links information.

7.8.2.6 `template<typename L> mln::trait::ch_value< L , bool >::ret scribo::make::text_blocks_image (const document< L > & doc, unsigned min_nlines)`

Create a mask of paragraph blocks.

Precondition

`doc` has_text() methods MUST return True.

7.8.2.7 `template<typename L> mln::trait::ch_value< L , bool >::ret scribo::make::text_components_image (const document< L > & doc)`

Create a binary image with text components only.

Precondition

`doc` has_text() methods MUST return True.

7.9 scribo::postprocessing Namespace Reference

Functions

- `template<typename I> mln::trait::concrete< I >::ret fill_object_holes (const Image< I > &input, float ratio)`
- `template<typename L> object_groups< L > fill_object_holes (const object_groups< L > &groups, unsigned min_size)`
- `template<typename L> void images_to_drop_capital (document< L > &doc)`

7.9.1 Detailed Description

Namespace of postprocessing routines.

7.9.2 Function Documentation

7.9.2.1 `template<typename I> mln::trait::concrete< I >::ret scribo::postprocessing::fill_object_holes (const Image< I > & input, float ratio) [inline]`

Fill-in object small holes.

7.9.2.2 `template<typename L> void scribo::postprocessing::images_to_drop_capital (document< L > & doc)`

Set type for specific images to Drop Capital component.

Parameters

<code>in</code>	<code>doc</code>	A document structure.
-----------------	------------------	-----------------------

7.10 scribo::preprocessing Namespace Reference

Functions

- `template<typename I >`
mln::trait::concrete< I >::ret [crop](#) (const **Image**< I > &input, const mln_box(I)&domain)
- `template<typename I >`
mln::trait::concrete< I >::ret [crop_without_localization](#) (const **Image**< I > &input, const mln_box(I)&domain)
- `template<typename I, typename N >`
mln::trait::concrete< I >::ret [denoise](#) (const **Image**< I > &input, const **Neighborhood**< N > &nbh, **unsigned** fg_min_card, **unsigned** bg_min_card)
- `template<typename I, typename N >`
mln::trait::concrete< I >::ret [denoise_bg](#) (const **Image**< I > &input, const **Neighborhood**< N > &nbh, **unsigned** min_card)
- `template<typename I, typename N >`
mln::trait::concrete< I >::ret [denoise_fg](#) (const **Image**< I > &input, const **Neighborhood**< N > &nbh, **unsigned** min_card)
- `template<typename I, typename J >`
mln::trait::concrete< I >::ret [deskew](#) (const **Image**< I > &crop_gl, const **Image**< I > &input_gl)
- `template<typename I >`
mln::trait::concrete< I >::ret [homogeneous_contrast](#) (const **Image**< I > &input, **unsigned** h)
- `template<typename I >`
mln::trait::concrete< I >::ret [homogeneous_contrast](#) (const **Image**< I > &input)
- `template<typename I >`
mln::trait::concrete< I >::ret [rotate_90](#) (const **Image**< I > &input, **bool** positive)
- `template<typename I >`
mln::trait::concrete< I >::ret [rotate_90](#) (const **Image**< I > &input)
- `template<typename I >`
mln::util::couple< typename
mln::trait::concrete< I >::ret,
typename **mln::trait::concrete**
< I >::ret > [split_bg_fg](#) (const **Image**< I > &input, **unsigned** lambda, **unsigned** delta)

7.10.1 Detailed Description

Namespace of preprocessing routines.

7.10.2 Function Documentation

- 7.10.2.1 `template<typename I > mln::trait::concrete< I >::ret scribo::preprocessing::crop (const Image< I > & input, const mln_box(I)& domain)`

crop an image preserving the localization.

Parameters

<i>in</i>	<i>input</i>	An image.
<i>in</i>	<i>domain</i>	A region of interest.

Returns

An image defined on the domain `domain` with the corresponding data copied from `input`.

7.10.2.2 `template<typename I> mln::trait::concrete<I>::ret scribo::preprocessing::crop_without_localization (const Image<I> & input, const mln_box(I)& domain)`

crop an image without preserving the localization.

Parameters

<code>in</code>	<code>input</code>	An image.
<code>in</code>	<code>domain</code>	A region of interest.

Returns

An image defined on a domain starting from the origin (`literal::origin`) and having the same size as `domain`. Its data corresponds to the data copied from `input` in the domain `domain`.

7.10.2.3 `template<typename I, typename N> mln::trait::concrete<I>::ret scribo::preprocessing::denoise (const Image<I> & input, const Neighborhood<N> & nbh, unsigned fg_min_card, unsigned bg_min_card)`

Denoise an image.

Denoising is performed both on the foreground and the background.

Parameters

<code>in</code>	<code>input</code>	A binary image. True for objects, False for background.
<code>in</code>	<code>nbh</code>	Neighborhood to use for denoising.
<code>in</code>	<code>fg_min_card</code>	Minimum component cardinality to not be considered as noise in the foreground.
<code>in</code>	<code>bg_min_card</code>	Minimum component cardinality to not be considered as noise in the foreground.

Returns

A binary image with the same domain as `input`. All small components have been removed.

7.10.2.4 `template<typename I, typename N> mln::trait::concrete<I>::ret scribo::preprocessing::denoise_bg (const Image<I> & input, const Neighborhood<N> & nbh, unsigned min_card)`

Denoise image background.

Parameters

<code>in</code>	<code>input</code>	A binary image. True for objects, False for background.
<code>in</code>	<code>nbh</code>	Neighborhood to use for denoising.
<code>in</code>	<code>min_card</code>	Minimum component cardinality to not be considered as noise.

Returns

A binary image with the same domain as `input`. All small components have been removed and merged with the background.

7.10.2.5 `template<typename I, typename N> mIn::trait::concrete<I>::ret scribo::preprocessing::denoise_fg (const Image<I> & input, const Neighborhood<N> & nbh, unsigned min_card)`

Denoise image foreground.

Parameters

in	<i>input</i>	A binary image. True for objects, False for background.
in	<i>nbh</i>	Neighborhood to use for denoising.
in	<i>min_card</i>	Minimum component cardinality to not be considered as noise.

Returns

A binary image with the same domain as *input*. All small components have been removed and merged with the background.

7.10.2.6 `template<typename I, typename J> mIn::trait::concrete<I>::ret scribo::preprocessing::deskew (const Image<I> & crop_gl, const Image<I> & input_gl)`

Deskew a region of interest.

Parameters

in	<i>crop_gl</i>	A gray-level image.
in	<i>input_gl</i>	A gray-level image.

Returns

A deskewed binary image.

Handles skew angles from -25 to +25 degrees.

crop_gl and *input_gl* must be 2D images and must be identical (e.g. only the value differs).

This algorithm is designed for images created from a region of interest (e.g. Not a full document).

7.10.2.7 `template<typename I> mIn::trait::concrete<I>::ret scribo::preprocessing::homogeneous_contrast (const Image<I> & input, unsigned h)`

Improve contrast homogeneity in an image.

Parameters

in	<i>input</i>	A gray-level image.
in	<i>h</i>	Height attribute value for leveling closing.

Returns

A gray-level image with better contrast homogeneity.

7.10.2.8 `template<typename I> mIn::trait::concrete<I>::ret scribo::preprocessing::homogeneous_contrast (const Image<I> & input)`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. Default height attribute value is set to 75.

7.10.2.9 `template<typename I> mIn::trait::concrete<I>::ret scribo::preprocessing::rotate_90 (const Image<I> & input, bool positive)`

Perform a +90/-90 degree rotation.

Parameters

<code>in</code>	<code>input</code>	An image.
<code>in</code>	<code>positive</code>	If set to true, performs a +90 degree rotation, -90 degree otherwise.

Returns

A rotated image.

7.10.2.10 `template<typename I> mIn::trait::concrete<I>::ret scribo::preprocessing::rotate_90 (const Image<I> & input)`

Performs a +90 degree rotation.

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts.

7.10.2.11 `template<typename I> mIn::util::couple<typename mIn::trait::concrete<I>::ret, typename mIn::trait::concrete<I>::ret> scribo::preprocessing::split_bg_fg (const Image<I> & input, unsigned lambda, unsigned delta)`

Split the background and the foreground.

Parameters

<code>in</code>	<code>input</code>	A color image.
<code>in</code>	<code>lambda</code>	Lambda used for morphological closing/opening.
<code>in</code>	<code>delta</code>	Max distance between values in closing and opening image.

Returns

A couple of color images. The first is the background and the second is the foreground.

7.11 scribo::primitive Namespace Reference

Namespaces

- namespace [extract](#)
- namespace [group](#)
- namespace [internal](#)
- namespace [link](#)

Functions

- `template<typename L> component_set<L> identify (const component_set<L> comps)`

7.11.1 Detailed Description

Namespace of primitive related routines.

7.12 scribo::primitive::extract Namespace Reference

Functions

- `template<typename L >`
mln::util::couple
`< component_set< L >, typename`
mln::trait::ch_value< L, **bool** >
`::ret > alignments (const document< L > &doc, float dmax_ratio, unsigned delta_pixel)`
- `template<typename I, typename V >`
mln::trait::ch_value< I, **bool** >
`::ret canvas (const Image< I > &input_, const mln::util::array< box< typename I::site > > &hlines_, const`
mln::util::array< **box**< typename I::site > > &vlines_, **unsigned** max_dist_lines)
- `template<typename I, typename N, typename V >`
mln::util::couple
`< mln::util::array< box`
`< typename I::site >`
`>, mln::util::array< box`
`< typename I::site > > > cells (const Image< I > &input, const Neighborhood< N > &nbh, const V`
`&label_type)`
- `template<typename I, typename J, typename N, typename V >`
[component_set](#)< typename
mln::trait::ch_value< I, V >
`::ret > components (const Image< I > &input, const Image< J > &binary_input, const Neighborhood< N`
`> &nbh, V &ncomponents, component::Type type=component::Undefined)`
- `template<typename I, typename N, typename V >`
[component_set](#)< typename
mln::trait::ch_value< I, V >
`::ret > components (const Image< I > &binary_input, const Neighborhood< N > &nbh, V &ncomponents,`
[component::Type](#) type=[component::Undefined](#))
- `template<typename I >`
mln::trait::concrete< I >::ret [horizontal_separators](#) (const [Image](#)< I > &input, **unsigned** line_length)
- `template<typename I, typename N, typename V, typename W >`
[component_set](#)< typename
mln::trait::ch_value< I, V >
`::ret > lines_discontinued (const Image< I > &input_, const Neighborhood< N > &nbh_, V &nlines, const`
[Window](#)< W > &win_, **unsigned** rank_k)
- `template<typename I, typename N, typename V >`
[component_set](#)< typename
mln::trait::ch_value< I, V >
`::ret > lines_h_discontinued (const Image< I > &input, const Neighborhood< N > &nbh, V &nlines, un-`
signed line_length, **unsigned** rank_k)
- `template<typename I >`
mln::trait::concrete< I >::ret [lines_h_pattern](#) (const [Image](#)< I > &input, **unsigned** length, **unsigned** delta)
- `template<typename I, typename N, typename V >`
[component_set](#)< typename
mln::trait::ch_value< I, V >
`::ret > lines_h_single (const Image< I > &input, const Neighborhood< N > &nbh, const V &nlines, un-`
signed min_line_length, **float** w_h_ratio)
- `template<typename L >`
[component_set](#)< L > [lines_h_single](#) (const [component_set](#)< L > &[components](#), **unsigned** min_line_length,
float w_h_ratio)

- `template<typename I, typename N, typename V >`
`component_set< typename`
`mln::trait::ch_value< I, V >`
`::ret > lines_h_thick_and_single (const Image< I > &input, const Neighborhood< N > &nbh, V &nlines,`
`unsigned min_line_length, float h_w_ratio)`
- `template<typename I >`
`mln::trait::concrete< I >::ret lines_h_thick_and_thin (const Image< I > &binary_image, unsigned length,`
`unsigned delta, float p_few=0.2, float p_enough=0.6, float ratio=8)`
- `template<typename I, typename W >`
`mln::trait::concrete< I >::ret lines_pattern (const Image< I > &input_, unsigned length, unsigned dir,`
`const Window< W > &win_)`
- `template<typename I, typename N, typename V, typename W >`
`component_set< typename`
`mln::trait::ch_value< I, V >`
`::ret > lines_thick (const Image< I > &input_, const Neighborhood< N > &nbh_, V &nlines, unsigned`
`line_length)`
- `template<typename I, typename N, typename V, typename W >`
`component_set< typename`
`mln::trait::ch_value< I, V >`
`::ret > lines_thick (const Image< I > &input_, const Neighborhood< N > &nbh_, V &nlines, const`
`Window< W > &win_)`
- `template<typename I, typename N, typename V >`
`component_set< typename`
`mln::trait::ch_value< I, V >`
`::ret > lines_v_discontinued (const Image< I > &input, const Neighborhood< N > &nbh, V &nlines, un-`
`signed line_length, unsigned rank_k)`
- `template<typename I >`
`mln::trait::concrete< I >::ret lines_v_pattern (const Image< I > &input, unsigned length, unsigned delta)`
- `template<typename I, typename N, typename V >`
`component_set< typename`
`mln::trait::ch_value< I, V >`
`::ret > lines_v_single (const Image< I > &input, const Neighborhood< N > &nbh, const V &nlines, un-`
`signed min_line_length, float h_w_ratio)`
- `template<typename L >`
`component_set< L > lines_v_single (const component_set< L > &components, unsigned min_line_length,`
`float h_w_ratio)`
- `template<typename I, typename N, typename V >`
`component_set< typename`
`mln::trait::ch_value< I, V >`
`::ret > lines_v_thick_and_single (const Image< I > &input, const Neighborhood< N > &nbh, V &nlines,`
`unsigned min_line_length, float h_w_ratio)`
- `template<typename L >`
`component_set< L > non_text (const document< L > &doc, unsigned nlines)`
- `template<typename L >`
`component_set< L > non_text_hdoc (const document< L > &doc, unsigned closing_size)`
- `template<typename L, typename I >`
`component_set< L > non_text_kmean (const document< L > &doc, const Image< I > &input)`
- `template<typename I >`
`mln::trait::concrete< I >::ret separators (const Image< I > &input, unsigned line_length)`
- `template<typename I >`
`mln::trait::concrete< I >::ret separators_nonvisible (const Image< I > &in_)`
- `template<typename I >`
`mln::trait::concrete< I >::ret vertical_separators (const Image< I > &input, unsigned line_length)`

7.12.1 Detailed Description

Namespace of primitive extraction related routines.

7.12.2 Function Documentation

7.12.2.1 `template<typename L > mln::util::couple<component_set<L>, typename mln::trait::ch_value< L , bool >::ret> scribo::primitive::extract::alignments (const document< L > & doc, float dmax_ratio, unsigned delta_pixel)`

Find page delimiters from tabstops and whitespaces.

Precondition

Separators should be removed from input document image .

Text in `doc` must be constructed from components grouped by lines with a very strict criterion in order to keep spaces between words and paragraphs. This first grouping is necessary to avoid false positive (e.g. inside the text blocks).

`doc` must have text (`doc.has_text()` returns true).

Internal description:

1) Build an image of line bboxes 2) For TOP and BOTTOM 2.a) Link bboxes. Links are validated only if :

- Alignment difference is less than `delta_pixel`.
- Bboxes are not too far
- No component is located at 5 pixels along the aligned side. 2.b) Invalidates groups if there are less than 3 links 2.c) Invalidates groups if a component is located at a specific distance from the aligned side.

This method handles skewed alignments and draw skew lines if possible. Examples :

```

|x      |x
| x      \x
|  x      \x
|   x      \x
          ->
|x
| x
|  x
|   x
|    x
|x

```

Here, the 'x' are aligned pair by pair but globally they are not. Here we cannot draw skewed lines without processing every links and look for that pattern. More over, in step 2.c alignment is validated by looking for other components in the supposed "whitespace area" from the aligned side. With such an alignment there are more chances that a component intersect with that line. Here, we would like to split links/alignement in two groups in order to get something like that :

```

\x
 \x
  \x
   /x
  /x
 /x

```

Parameters

in	<i>doc</i>	A document information with text lines.
in	<i>dmax_ratio</i>	The ratio used to compute the maximum lookup distance while linking up components.
in	<i>delta_pixel</i>	The maximum number of pixels allowed for alignment delta precision.

7.12.2.2 `template<typename I, typename V > mln::trait::ch_value< I, bool >::ret scribo::primitive::extract::canvas (const Image< I > & input_, const mln::util::array< box< typename I::site > > & hlines_, const mln::util::array< box< typename I::site > > & vlines_, unsigned max_dist_lines)`

Rebuild a table from its line bounding boxes.

Parameters

in	<i>input_</i>	A binary image.
in	<i>hlines_</i>	Vorizontal line bounding boxes.
in	<i>vlines_</i>	vertical line bounding boxes.
in	<i>max_dist_lines</i>	The maximum distance allowed between vertical and horizontal lines to connect them eachother.

Returns

The canvas as a binary image. canvas lines are set to true.

7.12.2.3 `template<typename I, typename N, typename V > mln::util::couple<mln::util::array<box<typename I::site> >, mln::util::array<box<typename I::site> > > scribo::primitive::extract::cells (const Image< I > & input, const Neighborhood< N > & nbh, const V & label_type)`

Extract canvas cells from a binary image.

Use arbitrary criterions.

Parameters

in	<i>input</i>	A binary image.
in	<i>nbh</i>	A neighborhood.
in, out	<i>label_type</i>	Type of the labeled image.

Returns

A list of cell bounding boxes.

7.12.2.4 `template<typename L > component_set<L> scribo::primitive::extract::non_text (const document< L > & doc, unsigned nlines)`

Extract non text components.

This method takes text localization into account and tries to learn the background colors to deduce the relevant non text components.

Parameters

in	<i>doc</i>	A document structure. Its must have paragraph information.
in	<i>nlines</i>	The number of lines needed in a paragraph to consider the latter during the background color learning.

Returns

A component set of non text components.

7.12.2.5 `template<typename L> component_set<L> scribo::primitive::extract::non_text_hdoc (const document< L > & doc, unsigned closing_size)`

Extract non text components.

Variant adapted for historical documents.

7.13 scribo::primitive::group Namespace Reference

Functions

- `template<typename L> component_set< L > apply (const object_groups< L > &groups)`
- `template<typename L> component_set< L > apply (const object_links< L > &links)`
- `template<typename L> object_groups< L > from_double_link (const object_links< L > &left_link, const object_links< L > &right_link)`
- `template<typename L> object_groups< L > from_double_link_any (const object_links< L > &left_link, const object_links< L > &right_link)`
- `template<typename L, typename G> object_groups< L > from_graph (const component_set< L > &comps, const Graph< G > &g_)`
- `template<typename L> object_groups< L > from_single_link (const object_links< L > &links)`
- `template<typename L> object_groups< L > regroup_left (const component_set< L > &components, const object_groups< L > &groups, unsigned dmax)`

7.13.1 Detailed Description

Namespace of primitive grouping related routines.

7.13.2 Function Documentation

7.13.2.1 `template<typename L> component_set<L> scribo::primitive::group::apply (const object_groups< L > & groups)`

Apply grouping in an object image.

Parameters

<i>groups</i>	An object group structure.
---------------	----------------------------

Returns

A copy of `components` with grouped components.

7.13.2.2 `template<typename L> component_set<L> scribo::primitive::group::apply (const object_links< L > & links)`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

7.13.2.3 `template<typename L> object_groups<L> scribo::primitive::group::from_double_link (const object_links< L > & left_link, const object_links< L > & right_link)`

Group components from left and right links information and validate These links. A link must exist in both ways to be validated.

Parameters

<code>in</code>	<code>left_link</code>	The left neighbor of each line of text.
<code>in</code>	<code>right_link</code>	The right neighbor of each line of text.

Returns

Return object groups information.

7.13.2.4 `template<typename L> object_groups<L> scribo::primitive::group::from_double_link_any (const object_links< L > & left_link, const object_links< L > & right_link)`

Group components from left and right links information and validate These links.

Parameters

<code>in</code>	<code>left_link</code>	The left neighbor of each line of text.
<code>in</code>	<code>right_link</code>	The right neighbor of each line of text.

Returns

Return object groups information.

7.13.2.5 `template<typename L , typename G> object_groups<L> scribo::primitive::group::from_graph (const component_set< L > & comps, const Graph< G > & g_)`

Group objects according to a graph of links.

Parameters

<code>in</code>	<code>comps</code>	A component set.
<code>in</code>	<code>g_</code>	The graph of object links.

Returns

Object groups information.

7.13.2.6 `template<typename L> object_groups<L> scribo::primitive::group::from_single_link (const object_links< L > & links)`

Link text components with their neighbor line if they have one.

Parameters

<code>in</code>	<code>links</code>	The neighbor line of each line.
-----------------	--------------------	---------------------------------

Returns

Object groups information.

7.14 scribo::primitive::internal Namespace Reference

Functions

- template<typename G , typename L >
void [find_graph_link](#) (**Graph**< G > &g_, const [component_set](#)< L > &comps, **unsigned** current_comp, **int** dmax, const typename L::site &c)
- template<typename F >
mln::util::couple< **bool**,
typename scribo_support_(F)::site > [find_link](#) ([Link_Functor](#)< F > &functor, **unsigned** current_object,
anchor::Type anchor)
- template<typename L >
unsigned [find_root](#) ([object_groups](#)< L > &parent, **unsigned** x)
- **bool** [have_link_valid](#) (const **mln::util::array**< **unsigned** > &left_link, const **mln::util::array**< **unsigned** >
&right_link, **unsigned** i)
- template<typename L >
bool [is_link_valid](#) (const [object_links](#)< L > &left_link, const [object_links](#)< L > &right_link, **unsigned** i)
- template<typename I , typename J >
mln::trait::concrete< I >::ret rd (const **Image**< I > &ima, const **Image**< J > &constraint)
- template<typename I , typename G >
void [update_graph_link](#) (const **Image**< I > &lb_, **Graph**< G > &g_, const typename I::site &p, const type-
name I::site &c, **unsigned** i, **int** dmax)

7.14.1 Detailed Description

Namespace of internal routines grouping text components.

7.14.2 Function Documentation

- 7.14.2.1 template<typename G , typename L > void scribo::primitive::internal::find_graph_link (**Graph**< G > &g_, const [component_set](#)< L > &comps, **unsigned** *current_comp*, **int** *dmax*, const typename L::site &c)

Find a neighbor of a component in a specific range if it exists.

Parameters

<i>g_</i>	The link graph.
<i>comps</i>	A component set.
<i>current_comp</i>	The current line being processed.
<i>dmax</i>	The maximum lookup distance.
<i>c</i>	Start point of the neighbor lookup.

- 7.14.2.2 template<typename F > **mln::util::couple**<**bool**, typename scribo_support_(F) ::site>
scribo::primitive::internal::find_link ([Link_Functor](#)< F > &functor, **unsigned** *current_object*, anchor::Type *anchor*)

Find the neighbor of a line of text if exists.

Parameters

<i>in</i> , <i>out</i>	<i>functor</i>	Functor used to compute the links. Stores the results.
<i>in</i>	<i>current_object</i>	Current object id.
<i>in</i>	<i>anchor</i>	The lookup anchor.

Returns

A couple. The first argument tells whether a valid link has been found, the second one is link anchor if exists.

7.14.2.3 `template<typename L> unsigned scribo::primitive::internal::find_root (object_groups< L> &parent, unsigned x)`

Find root in a parent array arrays.

7.14.2.4 `bool scribo::primitive::internal::have_link_valid (const mln::util::array< unsigned> &left_link, const mln::util::array< unsigned> &right_link, unsigned i)`

Tells whether a component have at least one valid link link.

Parameters

<i>in</i>	<i>left_link</i>	Left link of components.
<i>in</i>	<i>right_link</i>	Right link of components.
<i>in</i>	<i>i</i>	The component id.

Returns

True if the *i*-th component has at least one valid link.

7.14.2.5 `template<typename L> bool scribo::primitive::internal::is_link_valid (const object_links< L> &left_link, const object_links< L> &right_link, unsigned i)`

Validate a link from two different links.

Parameters

<i>in</i>	<i>left_link</i>	Left link of components.
<i>in</i>	<i>right_link</i>	Right link of components.
<i>in</i>	<i>i</i>	The component id.

Returns

True if the link is between the *i*-th component

7.14.2.6 `template<typename I, typename J> mln::trait::concrete< I>::ret scribo::primitive::internal::rd (const Image< I> &ima, const Image< J> &constraint)`

Tolerant constrained reconstruction algorithm.

7.14.2.7 `template<typename I, typename G> void scribo::primitive::internal::update_graph_link (const Image< I> &lbl, Graph< G> &g, const typename I::site &p, const typename I::site &c, unsigned i, int dmax)`

Update graph edges if a valid neighbor is found.

Parameters

in	<i>lbl_</i>	A label image.
in	<i>g_</i>	A graph.
in	<i>p</i>	A site of <i>lbl_</i> .
in	<i>c</i>	A site of <i>lbl_</i> .
in	<i>i</i>	A vertex id.
in	<i>dmax</i>	The maximum distance allowed to look for a neighbor.

7.15 scribo::primitive::link Namespace Reference

Functions

- template<typename F >
object_links< scribo_support(F)> compute (Link_Functor< F > &functor, anchor::Type anchor)
- template<typename F >
object_links< scribo_support(F)> compute (Link_Functor< F > &functor)
- template<typename F >
object_links< scribo_support(F)> compute_several (Link_Functor< F > &functor)
- template<typename L >
object_links< L > left (const component_set< L > &components, unsigned dmax)
- template<typename L >
mln::util::couple
< object_links< L >
, object_links< L > > left_right (const component_set< L > &components)
- template<typename L >
object_links< L > merge_double_link (const object_links< L > &left_link, const object_links< L > &right_link)
- template<typename L >
object_links< L > merge_double_link_closest_aligned (const object_links< L > &left, const object_links< L > &right, anchor::Type anchor_angle)
- template<typename L >
mln::util::graph with_graph (const component_set< L > &comps, unsigned neighb_max_distance)
- template<typename L , typename N >
util::couple< mln::util::graph,
typename mln::trait::concrete
< L >::ret > with_rag (const component_set< L > &comps, const Neighborhood< N > &nbh)
- template<typename L >
mln::util::graph with_several_graphes (const component_set< L > &comps, unsigned neighb_max_distance)
- template<typename L >
object_links< L > with_several_left_links (const component_set< L > &objects, unsigned neighb_max_distance)
- template<typename L >
object_links< L > with_several_left_links (const component_set< L > &comps)
- template<typename L >
object_links< L > with_several_right_closest_links (const component_set< L > &comps, unsigned neighb_max_distance)
- template<typename L >
object_links< L > with_several_right_closest_links (const component_set< L > &comps)
- template<typename L >
object_links< L > with_several_right_links (const component_set< L > &comps, unsigned neighb_max_distance)
- template<typename L >
object_links< L > with_several_right_links (const component_set< L > &comps)

- `template<typename L >`
`object_links< L > with_several_right_links_overlap` (const `object_image(L)&objects`, **unsigned** `neighb_max_distance`)
- `template<typename L >`
`object_links< L > with_several_right_links_overlap` (const `object_image(L)&objects`)
- `template<typename L >`
`object_links< L > with_single_down_link` (const `component_set< L > &comps`, **unsigned** `neighb_max_distance`, `anchor::Type anchor`)
- `template<typename L >`
`object_links< L > with_single_down_link` (const `component_set< L > &comps`, **unsigned** `neighb_max_distance`)
- `template<typename L >`
`object_links< L > with_single_down_link` (const `component_set< L > &comps`)
- `template<typename L >`
`object_links< L > with_single_left_link` (const `component_set< L > &components`, **unsigned** `neighb_max_distance`)
- `template<typename L >`
`object_links< L > with_single_left_link` (const `component_set< L > &components`)
- `template<typename L , typename F >`
`object_links< L > with_single_left_link_dmax_ratio` (const `component_set< L > &components`, const `DMax_Functor< F > &dmax_f`, `anchor::Type anchor`)
- `template<typename L , typename F >`
`object_links< L > with_single_left_link_dmax_ratio` (const `component_set< L > &components`, **float** `dmax_ratio`, `anchor::Type anchor`)
- `template<typename L >`
`object_links< L > with_single_left_link_dmax_ratio` (const `component_set< L > &components`, **float** `dmax_ratio`)
- `template<typename L , typename F >`
`object_links< L > with_single_left_link_dmax_ratio` (const `component_set< L > &components`)
- `template<typename L , typename F >`
`object_links< L > with_single_left_link_dmax_ratio_aligned` (const `component_set< L > &components`, const `DMax_Functor< F > &dmax_f`, **float** `min_angle`, **float** `max_angle`, `anchor::Type anchor`)
- `template<typename L >`
`object_links< L > with_single_left_link_dmax_ratio_aligned` (const `component_set< L > &components`, **float** `dmax_ratio`, **float** `min_angle`, **float** `max_angle`)
- `template<typename L >`
`object_links< L > with_single_left_link_dmax_ratio_aligned` (const `component_set< L > &components`)
- `template<typename L >`
`object_links< L > with_single_right_link` (const `component_set< L > &components`, **unsigned** `neighb_max_distance`, `anchor::Type anchor=anchor::MassCenter`)
- `template<typename L >`
`object_links< L > with_single_right_link` (const `component_set< L > &components`)
- `template<typename L >`
`object_links< L > with_single_right_link_bottom` (const `component_set< L > &components`, **unsigned** `neighb_max_distance`)
- `template<typename L >`
`object_links< L > with_single_right_link_bottom` (const `component_set< L > &components`)
- `template<typename L , typename F >`
`object_links< L > with_single_right_link_dmax_ratio` (const `component_set< L > &components`, const `DMax_Functor< F > &dmax_f`, `anchor::Type anchor`)
- `template<typename L >`
`object_links< L > with_single_right_link_dmax_ratio` (const `component_set< L > &components`, **float** `dmax_ratio`, `anchor::Type anchor`)
- `template<typename L >`
`object_links< L > with_single_right_link_dmax_ratio` (const `component_set< L > &components`, **float** `dmax_ratio`)

- `template<typename L >`
`object_links< L > with_single_right_link_dmax_ratio` (const `component_set< L >` &components)
- `template<typename L >`
`object_links< L > with_single_right_link_dmax_ratio_aligned` (const `component_set< L >` &components, **float** dmax_ratio, **float** min_angle, **float** max_angle, anchor::Type anchor)
- `template<typename L >`
`object_links< L > with_single_right_link_dmax_ratio_aligned` (const `component_set< L >` &components, **float** dmax_ratio, **float** min_angle, **float** max_angle)
- `template<typename L >`
`object_links< L > with_single_right_link_dmax_ratio_aligned` (const `component_set< L >` &components)
- `template<typename L >`
`object_links< L > with_single_right_link_top` (const `component_set< L >` &components, **unsigned** neighb_max_distance)
- `template<typename L >`
`object_links< L > with_single_right_link_top` (const `component_set< L >` &components)
- `template<typename L >`
`object_links< L > with_single_up_link` (const `component_set< L >` &comps, **unsigned** neighb_max_distance, anchor::Type anchor)
- `template<typename L >`
`object_links< L > with_single_up_link` (const `component_set< L >` &comps, **unsigned** neighb_max_distance)
- `template<typename L >`
`object_links< L > with_single_up_link` (const `component_set< L >` &comps)

7.15.1 Detailed Description

Namespace of primitive linking related routines.

7.15.2 Function Documentation

7.15.2.1 `template<typename F > object_links<scribo_support(F)> scribo::primitive::link::compute (Link_Functor< F > & functor, anchor::Type anchor)`

Compute links between objects according a given functor.

Parameters

<code>in, out</code>	<code>functor</code>	Linking policy.
<code>in</code>	<code>anchor</code>	Lookup anchor. Starts looking for neighbors from there.

Returns

Object links.

Functors must implement the following interface :

`bool is_potential_link_(unsigned current_object, const P& start_point, const P& p) const`

`bool valid_link_(unsigned current_object, const P& start_point, const P& p)`

`bool verify_link_criterion_(unsigned current_object, const P& start_point, const P& p)`

`void validate_link_(unsigned current_object, const P& start_point, const P& p, unsigned anchor)`

`void invalidate_link_(unsigned current_object, const P& start_point, const P& p, unsigned anchor)`

`void compute_next_site_(P& p)`

`const mIn_site(L)& start_point_(unsigned current_object, unsigned anchor)`

`void start_processing_object_(unsigned current_object)`

7.15.2.2 `template<typename F> object_links<scribo_support(F)> scribo::primitive::link::compute (Link_Functor< F> & functor)`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. The default anchor is set to `anchor::MassCenter`.

7.15.2.3 `template<typename F> object_links<scribo_support(F)> scribo::primitive::link::compute_several (Link_Functor< F> & functor)`

Compute_Several links between objects according a given functor.

Parameters

<code>in, out</code>	<code>functor</code>	Linking policy.
----------------------	----------------------	-----------------

Returns

Object links.

Functors must implement the following interface :

`bool verify_link_criterion_(unsigned current_object, const P& start_point, const P& p) const;`

`void start_processing_object_(unsigned current_object);`

`mln_site(L) start_point_(unsigned current_object, unsigned anchor);`

`void validate_link_(unsigned current_object, const P& start_point, const P& p, unsigned anchor); void invalidate_link_(unsigned current_object, const P& start_point, const P& p, unsigned anchor);`

`void initialize_link_(unsigned current_object); void finalize_link_(unsigned current_object);`

`bool is_potential_link(unsigned current_object, const P& start_point, const P& p) const`

`void compute_next_site(P& p)`

7.15.2.4 `template<typename L> object_links<L> scribo::primitive::link::merge_double_link (const object_links< L> & left_link, const object_links< L> & right_link)`

Validate and merge double link information. A link must exist in both ways to be validated.

Parameters

<code>in</code>	<code>left_link</code>	The left neighbor of each line of text.
<code>in</code>	<code>right_link</code>	The right neighbor of each line of text.

Returns

The merge of `left_link` and `right_link`.

7.15.2.5 `template<typename L> object_links<L> scribo::primitive::link::merge_double_link_closest_aligned (const object_links< L> & left, const object_links< L> & right, anchor::Type anchor_angle)`

Merge two object links data based on distance and angle.

Performs an 'OR' operation on the links according distance and angle criterion.

If a component has several incoming links, only the link for which the the object is the closest or the angle performed between the two bboxes is the lower, will be preserved.

7.15.2.6 `template<typename L> mln::util::graph scribo::primitive::link::with_graph (const component_set< L> & comps, unsigned neighb_max_distance)`

Construct the links between each line of text and store it as a graph. Look up for neighbors on the right of each box.

Parameters

in	<i>comps</i>	A component set.
in	<i>neighb_max_distance</i>	The maximum distance allowed to look for a neighbor.

Returns

A graph of relationship.

7.15.2.7 `template<typename L, typename N> util::couple<mln::util::graph, typename mln::trait::concrete< L>::ret> scribo::primitive::link::with_rag (const component_set< L> & comps, const Neighborhood< N> & nbh)`

Link components with a region adjacency graph.

7.15.2.8 `template<typename L> mln::util::graph scribo::primitive::link::with_several_graphes (const component_set< L> & comps, unsigned neighb_max_distance)`

Link character bounding boxes with several graphes.

Look up for neighbors on the left of each box.

7.15.2.9 `template<typename L> object_links<L> scribo::primitive::link::with_several_left_links (const component_set< L> & objects, unsigned neighb_max_distance) [inline]`

Map each character bounding box to its left bounding box neighbor if possible.

Iterate to the right but link boxes to the left.

Returns

object links data.

7.15.2.10 `template<typename L> object_links<L> scribo::primitive::link::with_several_left_links (const component_set< L> & comps) [inline]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

7.15.2.11 `template<typename L> object_links<L> scribo::primitive::link::with_several_right_closest_links (const component_set< L> & comps, unsigned neighb_max_distance) [inline]`

Map each character bounding box to its right bounding box neighbor if possible.

If there are several right neighbor, the closest one is chosen.

Returns

an `mln::util::array`. Map a bounding box to its right neighbor.

7.15.2.12 `template<typename L> object_links<L> scribo::primitive::link::with_several_right_closest_links (const component_set< L> & comps) [inline]`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts.

7.15.2.13 `template<typename L> object_links<L> scribo::primitive::link::with_several_right_links (const component_set< L> & comps, unsigned neighb_max_distance) [inline]`

Map each character bounding box to its right bounding box neighbor if possible.

Iterate to the right but link boxes to the right.

Returns

Object links.

7.15.2.14 `template<typename L> object_links<L> scribo::primitive::link::with_several_right_links (const component_set< L> & comps) [inline]`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts.

7.15.2.15 `template<typename L> object_links<L> scribo::primitive::link::with_several_right_links_overlap (const object_image(L)& objects, unsigned neighb_max_distance) [inline]`

Map each character bounding box to its right bounding box neighbor if possible.

Iterate to the right but link boxes to the right.

Returns

an `mln::util::array`. Map a bounding box to its right neighbor.

7.15.2.16 `template<typename L> object_links<L> scribo::primitive::link::with_several_right_links_overlap (const object_image(L)& objects) [inline]`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts.

7.15.2.17 `template<typename L> object_links<L> scribo::primitive::link::with_single_down_link (const component_set< L> & comps, unsigned neighb_max_distance, anchor::Type anchor) [inline]`

Link objects with their down neighbor if exists.

Parameters

in	<i>comps</i>	A component set.
in	<i>neighb_max_distance</i>	The maximum distance allowed to search a neighbor object.
in	<i>anchor</i>	The neighborhood lookup start point.

Returns

Object links data.

7.15.2.18 `template<typename L> object_links<L> scribo::primitive::link::with_single_down_link (const component_set< L> & comps, unsigned neighb_max_distance) [inline]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. Anchor type is set to anchor::MassCenter.

7.15.2.19 `template<typename L> object_links<L> scribo::primitive::link::with_single_down_link (const component_set< L> & comps) [inline]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. Max distance is set to mln_max(unsigned).

7.15.2.20 `template<typename L> object_links<L> scribo::primitive::link::with_single_left_link (const component_set< L> & components, unsigned neighb_max_distance) [inline]`

Link components with their left neighbor if exists.

Parameters

in	<i>components</i>	A component set.
in	<i>neighb_max_distance</i>	The maximum distance allowed to search a neighbor object.

Returns

Object links data.

7.15.2.21 `template<typename L> object_links<L> scribo::primitive::link::with_single_left_link (const component_set< L> & components) [inline]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. Max distance is set to mln_max(unsigned).

7.15.2.22 `template<typename L, typename F> object_links<L> scribo::primitive::link::with_single_left_link_dmax_ratio (const component_set< L> & components, const DMax_Functor< F> & dmax_f, anchor::Type anchor) [inline]`

Link components with their left neighbor if exists.

Parameters

in	<i>components</i>	A component set.
in	<i>anchor</i>	Starting point for the neighbor lookup.
in	<i>dmax_f</i>	DMax functor defining the maximum lookup distance.

Returns

Object links data.

Look for a neighbor until a maximum distance is reached. The maximum distance is defined thanks to a functor `dmax_f`.

7.15.2.23 `template<typename L , typename F > object_links<L> scribo::primitive::link::with_single_left_link_dmax_ratio (const component_set< L > & components, float dmax_ratio, anchor::Type anchor) [inline]`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. The default dmax functor is used ([internal::dmax_default](#)).

7.15.2.24 `template<typename L > object_links<L> scribo::primitive::link::with_single_left_link_dmax_ratio (const component_set< L > & components, float dmax_ratio) [inline]`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

7.15.2.25 `template<typename L , typename F > object_links<L> scribo::primitive::link::with_single_left_link_dmax_ratio (const component_set< L > & components) [inline]`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. The default dmax functor is used ([internal::dmax_default](#)) with `dmax_ratio` set to 3.

`anchor` is set to `anchor::MassCenter`.

7.15.2.26 `template<typename L , typename F > object_links<L> scribo::primitive::link::with_single_left_link_dmax_ratio_aligned (const component_set< L > & components, const DMax_Functor< F > & dmax_f, float min_angle, float max_angle, anchor::Type anchor) [inline]`

Link objects with their left neighbor if exists.

Parameters

in	<i>components</i>	A component set.
in	<i>dmax_f</i>	A function defining the maximum lookup distance.
in	<i>min_angle</i>	Minimum difference allowed for alignment angle.
in	<i>max_angle</i>	Maximum difference allowed for alignment angle.
in	<i>anchor</i>	Starting point for the neighbor lookup.

Returns

Object links data.

Look for a neighbor until a maximum distance defined by :

$$dmax = w / 2 + dmax_ratio * \max(h, w)$$

where `w` is the bounding box width and `h` the bounding box height.

7.15.2.27 `template<typename L > object_links<L> scribo::primitive::link::with_single_left_link_dmax_ratio_aligned (const component_set< L > & components, float dmax_ratio, float min_angle, float max_angle) [inline]`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. `anchor` is set to `MassCenter`.

`dmax_f` functor is set to [internal::dmax_default](#).

7.15.2.28 `template<typename L> object_links<L> scribo::primitive::link::with_single_left_link_dmax_ratio_aligned (const component_set< L> & components) [inline]`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. `dmax_ratio` is set to 3.

anchor is set to `MassCenter`.

7.15.2.29 `template<typename L> object_links<L> scribo::primitive::link::with_single_right_link (const component_set< L> & components, unsigned neighb_max_distance, anchor::Type anchor = anchor::MassCenter) [inline]`

Link components with their right neighbor if exists.

Lookup startup point is the object mass center.

Parameters

in	<i>components</i>	An object image.
in	<i>neighb_max_distance</i>	The maximum distance allowed to search a neighbor object.
in	<i>anchor</i>	Anchor from where the neighbor lookup is performed.

Returns

Object links data.

7.15.2.30 `template<typename L> object_links<L> scribo::primitive::link::with_single_right_link (const component_set< L> & components) [inline]`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. Max distance is set to `mln_max(unsigned)`.

7.15.2.31 `template<typename L> object_links<L> scribo::primitive::link::with_single_right_link_bottom (const component_set< L> & components, unsigned neighb_max_distance) [inline]`

Link components with their right neighbor if exists.

Lookup startup point is the object bottom center.

Parameters

in	<i>components</i>	A component set.
in	<i>neighb_max_distance</i>	The maximum distance allowed to search a neighbor object.

Returns

Object links data.

7.15.2.32 `template<typename L> object_links<L> scribo::primitive::link::with_single_right_link_bottom (const component_set< L> & components) [inline]`

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. Max distance is set to `mln_max(unsigned)`.

7.15.2.33 `template<typename L , typename F > object_links<L> scribo::primitive::link::with_single_right_link_dmax_ratio (const component_set< L > & components, const DMax_Functor< F > & dmax_f, anchor::Type anchor) [inline]`

Link objects with their right neighbor if exists.

Parameters

in	<i>components</i>	A component set.
in	<i>anchor</i>	Starting point for the neighbor lookup.
in	<i>dmax_f</i>	DMax functor defining the maximum lookup distance.

Returns

Object links data.

Look for a neighbor until a maximum distance is reached. The maximum distance is defined thanks to a functor `dmax_f`.

7.15.2.34 `template<typename L > object_links<L> scribo::primitive::link::with_single_right_link_dmax_ratio (const component_set< L > & components, float dmax_ratio, anchor::Type anchor) [inline]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `dmax_f` is set to [internal::dmax_default](#).

7.15.2.35 `template<typename L > object_links<L> scribo::primitive::link::with_single_right_link_dmax_ratio (const component_set< L > & components, float dmax_ratio) [inline]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `MassCenter`.

7.15.2.36 `template<typename L > object_links<L> scribo::primitive::link::with_single_right_link_dmax_ratio (const component_set< L > & components) [inline]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `dmax_ratio` is set to 3.

`anchor` is set to `MassCenter`.

7.15.2.37 `template<typename L > object_links<L> scribo::primitive::link::with_single_right_link_dmax_ratio_aligned (const component_set< L > & components, float dmax_ratio, float min_angle, float max_angle, anchor::Type anchor) [inline]`

Link objects with their right neighbor if exists.

Parameters

in	<i>components</i>	A component set.
in	<i>dmax_ratio</i>	Size ratio defining the maximum lookup distance.
in	<i>min_angle</i>	Minimum difference allowed for alignment angle.
in	<i>max_angle</i>	Maximum difference allowed for alignment angle.
in	<i>anchor</i>	Starting point for the neighbor lookup.

Returns

Object links data.

Look for a neighbor until a maximum distance defined by :

$$dmax = w / 2 + dmax_ratio * \max(h, w)$$

where w is the bounding box width and h the bounding box height.

7.15.2.38 `template<typename L> object_links<L> scribo::primitive::link::with_single_right_link_dmax_ratio_aligned (const component_set< L> & components, float dmax_ratio, float min_angle, float max_angle) [inline]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.anchor is set to MassCenter.

7.15.2.39 `template<typename L> object_links<L> scribo::primitive::link::with_single_right_link_dmax_ratio_aligned (const component_set< L> & components) [inline]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.dmax_ratio is set to 3.

anchor is set to MassCenter.

7.15.2.40 `template<typename L> object_links<L> scribo::primitive::link::with_single_right_link_top (const component_set< L> & components, unsigned neighb_max_distance) [inline]`

Link components with their right neighbor if exists.

Lookup startup point is the object top center.

Parameters

in	<i>components</i>	A component set.
in	<i>neighb_max_distance</i>	The maximum distance allowed to seach a neighbor object.

Returns

Object links data.

7.15.2.41 `template<typename L> object_links<L> scribo::primitive::link::with_single_right_link_top (const component_set< L> & components) [inline]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.Max distance is set to mln_max(unsigned).

7.15.2.42 `template<typename L> object_links<L> scribo::primitive::link::with_single_up_link (const component_set< L> & comps, unsigned neighb_max_distance, anchor::Type anchor) [inline]`

Link components with their up neighbor if exists.

Parameters

in	<i>comps</i>	A component set.
in	<i>neighb_max_distance</i>	The maximum distance allowed to seach a neighbor object.
in	<i>anchor</i>	The neighborhood lookup start point.

Returns

Object links data.

7.15.2.43 `template<typename L> object_links<L> scribo::primitive::link::with_single_up_link (const component_set< L > & comps, unsigned neighb_max_distance) [inline]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. Anchor type is set to `anchor::Center`.

7.15.2.44 `template<typename L> object_links<L> scribo::primitive::link::with_single_up_link (const component_set< L > & comps) [inline]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. Max distance is set to `mln_max(unsigned)`.

7.16 scribo::table Namespace Reference

Namespaces

- namespace [internal](#)

Functions

- `template<typename I, typename L>
mln::util::array< int > align_lines_horizontally (const Image< I > &input, const component_set< L > &lines, component_set< L > &aligned_lines, unsigned max_alignment_diff)`
- `template<typename I>
mln::util::array< int > align_lines_vertically (const Image< I > &input, mln::util::array< box< typename I::site > > &line_bboxes, unsigned max_alignment_diff)`
- `template<typename I>
void connect_horizontal_lines (const mln::util::array< int > &aligned_cols, mln::util::couple< mln::util::array< box< typename I::site > >, mln::util::array< box< typename I::site > > > &tablebbboxes, const Image< I > &input, unsigned max_distance)`
- `template<typename I>
void connect_vertical_lines (const mln::util::array< int > &aligned_rows, mln::util::couple< mln::util::array< box< typename I::site > >, mln::util::array< box< typename I::site > > > &tablebbboxes, const Image< I > &input, unsigned max_distance)`
- `template<typename I, typename L>
mln::trait::concrete< I >::ret erase (const Image< I > &input, const component_set< L > &hlines, const component_set< L > &vlines)`
- `template<typename I, typename V>
mln::util::couple< typename
mln::trait::ch_value< I, V >
::ret, mln::util::couple
< mln::util::array< box
< typename I::site >
>, mln::util::array< box
< typename I::site > > > > extract (const Image< I > &input_, V &ncells)`
- `template<typename I, typename L>
mln::util::couple< L,
mln::util::couple
< component_set< L >
, component_set< L > > > rebuild (const Image< I > &input, const component_set< L > &vlines, const component_set< L > &hlines, unsigned max_dist_lines, typename L::value &ncells)`

- `template<typename I >`
`void repair_horizontal_lines (const Image< I > &input, mln::util::couple< mln::util::array< box< typename I::site > >, mln::util::array< box< typename I::site > > > &tablebbboxes, unsigned max_discontinuity)`
- `template<typename I >`
`void repair_vertical_lines (const Image< I > &input, mln::util::couple< mln::util::array< box< typename I::site > >, mln::util::array< box< typename I::site > > > &tablebbboxes, unsigned max_discontinuity)`

7.16.1 Detailed Description

Namespace of routines working on tables.

7.16.2 Function Documentation

- 7.16.2.1 `template<typename I, typename L > mln::util::array<int> scribo::table::align_lines_horizontally (const Image< I > &input, const component_set< L > &lines, component_set< L > &aligned_lines, unsigned max_alignment_diff)`

Align line bounding boxes horizontally.

Parameters

in	<i>input</i>	Image from which the line bboxes are extracted from.
in	<i>lines</i>	Component set corresponding to table lines.
in, out	<i>aligned_lines</i>	Component set where aligned table lines are stored.
in	<i>max_alignment_diff</i>	max space between two lines to consider they are potentially on the same line.

Returns

A list of the resulting aligned rows. Each integer is actually a row number.

- 7.16.2.2 `template<typename I > mln::util::array<int> scribo::table::align_lines_vertically (const Image< I > &input, mln::util::array< box< typename I::site > > &line_bboxes, unsigned max_alignment_diff)`

Align line bounding boxes vertically.

Parameters

in	<i>input</i>	Image from which the line bboxes are extracted from.
in, out	<i>line_bboxes</i>	vertical lines bounding boxes.
in	<i>max_alignment_diff</i>	max space between two lines to consider they are potentially on the same line.

Returns

A list of the resulting aligned cols. Each integer is actually a col number.

- 7.16.2.3 `template<typename I > void scribo::table::connect_horizontal_lines (const mln::util::array< int > &aligned_cols, mln::util::couple< mln::util::array< box< typename I::site > >, mln::util::array< box< typename I::site > > > &tablebbboxes, const Image< I > &input, unsigned max_distance)`

Connect horizontal lines with the new aligned columns.

Parameters

in	<i>aligned_cols</i>	a list of new aligned cols.
in, out	<i>tablebbboxes</i>	the vertical and horizontal lines bounding boxes.
in	<i>input</i>	The image from where the lines are extracted.
in	<i>max_distance</i>	max distance allowed between a vertical and horizontal lines.

7.16.2.4 `template<typename I> void scribo::table::connect_vertical_lines (const mln::util::array< int > & aligned_rows, mln::util::couple< mln::util::array< box< typename I::site >>, mln::util::array< box< typename I::site >>> & tablebbboxes, const Image< I > & input, unsigned max_distance)`

Connect vertical lines with the new aligned rows.

Parameters

in	<i>aligned_rows</i>	a list of new aligned rows.
in, out	<i>tablebbboxes</i>	the vertical and horizontal lines bounding boxes.
in	<i>input</i>	The image from where the lines are extracted.
in	<i>max_distance</i>	max distance allowed between a vertical and horizontal lines.

7.16.2.5 `template<typename I, typename L> mln::trait::concrete< I >::ret scribo::table::erase (const Image< I > & input, const component_set< L > & hlines, const component_set< L > & vlines)`

Erase vertical and horizontal lines from an image.

Parameters

in	<i>input</i>	A binary image from which the table lines are extracted.
in	<i>hlines</i>	A component set with horizontal lines.
in	<i>vlines</i>	A component set with vertical lines.

Returns

A copy of *in* where the table lines are removed.

7.16.2.6 `template<typename I, typename V> mln::util::couple<typename mln::trait::ch_value< I, V >::ret, mln::util::couple<mln::util::array<box<typename I::site>>, mln::util::array<box<typename I::site>>>> & scribo::table::extract (const Image< I > & input, V & ncells)`

Extract tables from a binary image.

Use arbitrary criterions.

7.16.2.7 `template<typename I, typename L> mln::util::couple<L, mln::util::couple<component_set<L>, component_set<L>>> & scribo::table::rebuild (const Image< I > & input, const component_set< L > & vlines, const component_set< L > & hlines, unsigned max_dist_lines, typename L::value & ncells)`

Rebuild a table from its line bounding boxes.

Parameters

in	<i>input</i>	A binary image.
in	<i>vlines</i>	Component set corresponding to vertical lines.
in	<i>hlines</i>	Component set corresponding to horizontal lines.
in	<i>max_dist_lines</i>	The maximum distance allowed between vertical and horizontal lines to connect them eachother.
out	<i>ncells</i>	Store the number of cells found in the rebuilt tables.

Returns

A couple. The first argument is a label image in which each table cell is labeled. The second argument are the aligned and connected table line bounding boxes.

7.16.2.8 `template<typename I> void scribo::table::repair_horizontal_lines (const Image< I> & input, mIn::util::couple< mIn::util::array< box< typename I::site>>, mIn::util::array< box< typename I::site>>> & tableboxes, unsigned max_discontinuity)`

Repair horizontal lines which have small discontinuities.

— — ==> ————

Parameters

<code>in</code>	<code>input</code>	Image from which the table bounding boxes are extracted.
<code>in, out</code>	<code>tableboxes</code>	Table line bounding boxes.
<code>in</code>	<code>max_ - discontinuity</code>	Repair discontinuity which are smaller than this value.

7.16.2.9 `template<typename I> void scribo::table::repair_vertical_lines (const Image< I> & input, mIn::util::couple< mIn::util::array< box< typename I::site>>, mIn::util::array< box< typename I::site>>> & tableboxes, unsigned max_discontinuity)`

Repair vertical lines which have small discontinuities.

||
||
==> |
||
||

Parameters

<code>in</code>	<code>input</code>	Image from which the table bounding boxes are extracted.
<code>in, out</code>	<code>tableboxes</code>	Table line bounding boxes.
<code>in</code>	<code>max_ - discontinuity</code>	Repair discontinuity which are smaller than this value.

7.17 scribo::table::internal Namespace Reference

Functions

- `template<typename L> mIn::util::array< int> align_lines (unsigned nsites, int min_coord, int max_coord, const component_set< L> &lines, component_set< L> &aligned_lines, unsigned dim, unsigned max_alignment_diff)`
- `template<typename P> void connect_lines (const mIn::util::array< int> &aligned_lines, mIn::util::array< box< P>> &boxes, unsigned dim, unsigned dim_size, unsigned max_distance)`
- `template<unsigned axis, typename I> void repair_lines (const Image< I> &input, mIn::util::array< box< typename I::site>> &tableboxes, unsigned max_discontinuity)`

7.17.1 Detailed Description

Namespace of internal routines working on tables.

7.17.2 Function Documentation

7.17.2.1 `template<typename L > mIn::util::array<int> scribo::table::internal::align_lines (unsigned nsites, int min_coord, int max_coord, const component_set< L > & lines, component_set< L > & aligned_lines, unsigned dim, unsigned max_alignment_diff)`

Align table lines bboxes according to a given dimension.

Parameters

in	<i>nsites</i>	Number of sites in the given dimension <i>dim</i> .
in	<i>min_coord</i>	The minimal coordinate in the dimension <i>dim</i> .
in	<i>max_coord</i>	The maximal coordinate in the dimension <i>dim</i> .
in	<i>lines</i>	The line components.
out	<i>aligned_lines</i>	The components of the aligned lines.
in	<i>dim</i>	The dimension according which the lines are aligned.
in	<i>max_alignment_diff</i>	Maximum alignment difference.

Returns

A list of the resulting aligned cols. Each integer is actually a col number.

7.17.2.2 `template<typename P > void scribo::table::internal::connect_lines (const mIn::util::array< int > & aligned_lines, mIn::util::array< box< P > > & boxes, unsigned dim, unsigned dim_size, unsigned max_distance)`

Connect vertical and horizontal lines if they are close to each other.

—> ||||

FIXME: doc arguments.

7.17.2.3 `template<unsigned axis, typename I > void scribo::table::internal::repair_lines (const Image< I > & input, mIn::util::array< box< typename I::site > > & tableboxes, unsigned max_discontinuity)`

Repair lines which have small discontinuities.

FIXME: buggy. Sometimes few lines move or shrink!

Parameters

in	<i>input</i>	A binary image.
in, out	<i>tableboxes</i>	Vertical or horizontal lines.
in	<i>max_discontinuity</i>	The maximum discontinuity length which can be repaired.

7.18 scribo::text Namespace Reference

Functions

- `template<typename L , typename I >`

- mln::trait::concrete**< I >::ret **clean** (const **line_info**< L > &line, const **Image**< I > &input_)
- template<typename L , typename I >
void **clean_inplace** (const **line_info**< L > &line, **Image**< I > &input)
 - template<typename I , typename N >
line_set< typename
mln::trait::ch_value< I,
scribo::def::lbl_type >::ret > **extract_lines** (const **Image**< I > &input, const **Neighborhood**< N > &nbh,
const typename **mln::trait::ch_value**< I, **bool** >::ret &separators)
 - template<typename I , typename N >
line_set< typename
mln::trait::ch_value< I,
scribo::def::lbl_type >::ret > **extract_lines** (const **Image**< I > &input, const **Neighborhood**< N > &nbh)
 - template<typename L >
line_set< typename
mln::trait::ch_value< L,
scribo::def::lbl_type >::ret > **extract_lines** (const **component_set**< L > &input)
 - template<typename I , typename J , typename N >
line_set< typename
mln::trait::ch_value< I,
scribo::def::lbl_type >::ret > **extract_lines_with_features** (const **Image**< I > &input, const **Image**< J >
&input_binary, const **Neighborhood**< N > &nbh, const typename **mln::trait::ch_value**< I, **bool** >::ret
&separators)
 - template<typename I , typename J , typename N >
line_set< typename
mln::trait::ch_value< I,
scribo::def::lbl_type >::ret > **extract_lines_with_features** (const **Image**< I > &input, const **Image**< J >
&input_binary, const **Neighborhood**< N > &nbh)
 - template<typename I , typename N >
line_set< typename
mln::trait::ch_value< I,
scribo::def::lbl_type >::ret > **extract_lines_wo_merge** (const **Image**< I > &input, const **Neighborhood**<
N > &nbh, const typename **mln::trait::ch_value**< I, **bool** >::ret &separators)
 - template<typename I , typename N >
line_set< typename
mln::trait::ch_value< I,
scribo::def::lbl_type >::ret > **extract_lines_wo_merge** (const **Image**< I > &input, const **Neighborhood**<
N > &nbh)
 - template<typename L , typename N >
line_set< typename
mln::trait::ch_value< L,
scribo::def::lbl_type >::ret > **extract_lines_wo_merge** (const **document**< L > &doc, const **Neighborhood**<
N > &nbh)
 - template<typename L , typename N >
line_set< typename
mln::trait::ch_value< L,
scribo::def::lbl_type >::ret > **extract_lines_wo_merge** (const **document**< L > &doc, const **Neighborhood**<
N > &nbh, const typename **mln::trait::ch_value**< L, **bool** >::ret &separators)
 - template<typename L >
paragraph_set< L > **extract_paragraphs** (**line_set**< L > &lines, const **image2d**< **bool** > &input)
 - template<typename L >
paragraph_set< L > **extract_paragraphs_hdoc** (**line_set**< L > &lines, const **image2d**< **bool** > &input)
 - template<typename L >
line_links< L > **link_lines** (const **line_set**< L > &lines)
 - template<typename L >
line_set< L > **look_like_text_lines** (const **scribo::line_set**< L > &l)
 - template<typename L >
void **look_like_text_lines_inplace** (**scribo::line_set**< L > &line)

- template<typename L >
 line_set< L > merging (const scribo::line_set< L > &lines)
- template<typename L >
 line_set< L > merging_hdoc (const scribo::line_set< L > &lines)
- template<typename L >
 mIn::trait::concrete< L >::ret paragraphs_closing (const paragraph_set< L > &parset)
- template<typename L >
 void recognition (line_set< L > &lines, const char *language)
- template<typename I >
 void recognition (const Image< I > &line, const char *language, const std::string &output_file=std::string())

7.18.1 Detailed Description

Namespace of routines working on text components.

7.18.2 Function Documentation

7.18.2.1 template<typename L , typename I > mIn::trait::concrete< I >::ret scribo::text::clean (const line_info< L > & line, const Image< I > & input_)

Improve quality of an image with text.

Parameters

in	line	Line info providing statistics about the text in the corresponding image input.
in	input_	A binary image. Object are set to 'false' and backgroud to 'true'.

Returns

An image. The text have better quality.

7.18.2.2 template<typename L , typename I > void scribo::text::clean_inplace (const line_info< L > & line, Image< I > & input)

Improve quality of an image with text.

Parameters

in	line	Line info providing statistics about the text in the corresponding image input.
in, out	input	A binary image. Object are set to 'false' and backgroud to 'true'.

7.18.2.3 template<typename L > line_set<L> scribo::text::look_like_text_lines (const scribo::line_set< L > & l)

Set line type to line::Text according to criterion.

7.18.2.4 template<typename L > void scribo::text::look_like_text_lines_inplace (scribo::line_set< L > & line)

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. Inplace version.

Chapter 8

Class Documentation

8.1 `cluster_stats< T >` Class Template Reference

Public Member Functions

- **`cluster_stats`** (const **unsigned** size)
- **`T max`** ()
- **`T mean`** ()
- **`T median`** ()
- **`T min`** ()
- **`unsigned nelements`** ()
- **`T operator[]`** (const **unsigned** index)
- void **`reset`** ()
- void **`sort`** ()
- **`T standard_deviation`** ()
- void **`take`** (const **T** &value)
- **`T variance`** ()

8.1.1 Detailed Description

`template<typename T>class cluster_stats< T >`

Definition at line 31 of file stats.hh.

8.2 `compare_values< T >` Struct Template Reference

Public Member Functions

- **`bool operator()`** (const **T** &lhs, const **T** &rhs)

8.2.1 Detailed Description

`template<typename T>struct compare_values< T >`

Definition at line 16 of file stats.hh.

8.3 mln::info Struct Reference

Public Member Functions

- **int** **height** () const
- void **init** (**unsigned** p, **int** row, **int** col)
- void **update** (info &r)
- **int** **width** () const

Public Attributes

- **unsigned** card
- **float** col_sum
- **point2d** p_max
- **point2d** p_min
- **unsigned** parent
- **float** row_sum

8.3.1 Detailed Description

Definition at line 71 of file components.hh.

8.4 scribo::binarization::internal::niblack_formula Struct Reference

Public Member Functions

- **double** **operator**() (const **double** m_x_y, const **double** s_x_y, const **double** K) const
- **double** **operator**() (const **double** m_x_y, const **double** s_x_y) const

8.4.1 Detailed Description

Definition at line 52 of file niblack_formula.hh.

8.4.2 Member Function Documentation

8.4.2.1 **double** scribo::binarization::internal::niblack_formula::operator() (const **double** m_x_y, const **double** s_x_y, const **double** K) const

compute a threshold using Niblack's formula.

Parameters

in	<i>m_x_y</i>	Mean value.
in	<i>s_x_y</i>	Standard deviation.
in	<i>K</i>	Control the threshold value in the local window. The higher, the lower the threshold form the local mean $m(x, y)$.

Returns

A threshold.

8.4.2.2 `double scribo::binarization::internal::niblack_formula::operator() (const double m_x_y, const double s_x_y) const`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. *K* is set to -0.2.

8.5 scribo::binarization::internal::niblack_functor< I > Struct Template Reference

Public Types

- enum { **step** = 1 }

Public Member Functions

- **niblack_functor** (const **Image**< I > &input, **double** K)
- void **end_of_row** (int)
- void **exec** (**double** mean, **double** stddev)
- void **finalize** ()
- void **init** ()

Public Attributes

- [scribo::binarization::internal::niblack_formula](#) **formula_**
- const I **input**
- **double** **K_**
- **unsigned** **next_line**
- **mln::trait::ch_value**< I, **bool** >
::ret **output**
- const I::value * **pi**
- **bool** * **po**

8.5.1 Detailed Description

`template<typename I>struct scribo::binarization::internal::niblack_functor< I >`

Definition at line 57 of file `niblack_functor.hh`.

8.6 scribo::binarization::internal::niblack_functor_fast< I > Struct Template Reference

Public Types

- enum { **step** = 3 }

Public Member Functions

- **niblack_functor_fast** (const **Image**< I > &input, **double** K)
- void **end_of_row** (int)
- void **exec** (**double** mean, **double** stddev)
- void **finalize** ()
- void **init** ()

Public Attributes

- [scribo::binarization::internal::niblack_formula](#) **formula_**
- const I **input**
- double **K_**
- unsigned **next_line3**
- unsigned **offset1**
- unsigned **offset2**
- **mln::trait::ch_value**< I, bool >
::ret **output**
- const I::value * **pi**
- bool * **po**

8.6.1 Detailed Description

template<typename I>struct scribo::binarization::internal::niblack_functor_fast< I >

Definition at line 58 of file niblack_functor_fast.hh.

8.7 scribo::binarization::internal::niblack_threshold_functor< I > Struct Template Reference

Public Types

- enum { **step** = 3 }
- typedef **mln::trait::concrete**
< I >::ret **th_t**

Public Member Functions

- **niblack_threshold_functor** (const **Image**< I > &input, double K)
- void **end_of_row** (int)
- void **exec** (double mean, double stddev)
- void **finalize** ()
- void **init** ()

Public Attributes

- [scribo::binarization::internal::niblack_formula](#) **formula_**
- **mln::trait::concrete**< I >::ret **input**
- double **K_**
- unsigned **next_line3**
- unsigned **offset1**
- unsigned **offset2**
- **th_t** **output**
- I::value * **po**

8.7.1 Detailed Description

template<typename I>struct scribo::binarization::internal::niblack_threshold_functor< I >

Definition at line 58 of file niblack_threshold_functor.hh.

8.8 scribo::binarization::internal::sauvola_formula Struct Reference

Public Member Functions

- **double** [operator\(\)](#) (const **double** *m_x_y*, const **double** *s_x_y*, const **double** *K*, const **double** *R*) const
- **double** [operator\(\)](#) (const **double** *m_x_y*, const **double** *s_x_y*) const

8.8.1 Detailed Description

Definition at line 63 of file `sauvola_formula.hh`.

8.8.2 Member Function Documentation

8.8.2.1 **double** `scribo::binarization::internal::sauvola_formula::operator()` (const **double** *m_x_y*, const **double** *s_x_y*, const **double** *K*, const **double** *R*) const

Compute a threshold using Sauvola's formula.

Parameters

in	<i>m_x_y</i>	Mean value.
in	<i>s_x_y</i>	Standard deviation.
in	<i>K</i>	Controls the threshold value in the local window. The higher, the lower the threshold form the local mean $m(x, y)$.
in	<i>R</i>	Maximum value of the standard deviation (128 for grayscale documents).

Returns

A threshold.

8.8.2.2 **double** `scribo::binarization::internal::sauvola_formula::operator()` (const **double** *m_x_y*, const **double** *s_x_y*) const

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. *K* is set to 0.34 and *R* to 128.

8.9 scribo::binarization::internal::sauvola_funcutor< I > Struct Template Reference

Public Types

- enum { **step** = 3 }

Public Member Functions

- **sauvola_funcutor** (const **Image**< I > &input, **double** *K*, **double** *R*)
- void **end_of_row** (int)
- void **exec** (**double** mean, **double** stddev)
- void **finalize** ()
- void **init** ()

Public Attributes

- [scribo::binarization::internal::sauvola_formula](#) **formula_**
- const I **input**
- **double** **K_**
- **unsigned** **next_line3**
- **unsigned** **offset1**
- **unsigned** **offset2**
- **mln::trait::ch_value**< I, **bool** >
::ret **output**
- const I::value * **pi**
- **bool** * **po**
- **double** **R_**

8.9.1 Detailed Description

template<typename I>struct scribo::binarization::internal::sauvola_funcutor< I >

Definition at line 58 of file sauvola_funcutor.hh.

8.10 scribo::binarization::internal::sauvola_ms_funcutor< I > Struct Template Reference

Public Member Functions

- **sauvola_ms_funcutor** (const I &input, **double** R, const **image2d**< **value::int_u8** > &e_2, **unsigned** i, **unsigned** q)
- void **end_of_row** (**int** row)
- void **exec** (**double** mean, **double** stddev)
- void **finalize** ()
- void **init** ()
- **mln_fwd_pixter** (const I) **pxl**

Public Attributes

- **image2d**< **unsigned** > **card**
- **mln::util::array**< **int** > **dp**
- const **image2d**< **value::int_u8** > & **e_2**
- [sauvola_formula](#) **formula_**
- **unsigned** **i**
- **int** **i_**
- const I & **input**
- **double** **K_**
- **image2d**< **bool** > **msk**
- **unsigned** **n_nbhs**
- **image2d**< **unsigned** > **parent**
- **unsigned** **q**
- **double** **R_**
- **double** **res**
- **image2d**< **value::int_u8** > **t_sub**

8.10.1 Detailed Description

template<typename I>struct scribo::binarization::internal::sauvola_ms_functor< I >

Definition at line 66 of file sauvola_ms_functor.hh.

8.11 scribo::binarization::internal::sauvola_threshold_functor< I > Struct Template Reference

Public Types

- enum { **step** = 3 }
- typedef **mln::trait::concrete**
< I >::ret th_t

Public Member Functions

- **sauvola_threshold_functor** (const **Image**< I > &input, **double** K, **double** R)
- void **end_of_row** (int)
- void **exec** (**double** mean, **double** stddev)
- void **finalize** ()
- void **init** ()

Public Attributes

- scribo::binarization::internal::sauvola_formula **formula_**
- **mln::trait::concrete**< I >::ret input
- **double** K_
- **unsigned** next_line3
- **unsigned** offset1
- **unsigned** offset2
- th_t output
- **I::value** * po
- **double** R_

8.11.1 Detailed Description

template<typename I>struct scribo::binarization::internal::sauvola_threshold_functor< I >

Definition at line 58 of file sauvola_threshold_functor.hh.

8.12 scribo::binarization::internal::singh_formula< V > Struct Template Reference

Public Member Functions

- **double** **operator()** (const V &v, const **double** m_x_y, const **double** K) const
- **double** **operator()** (const V &v, const **double** m_x_y) const

8.12.1 Detailed Description

template<typename V>struct scribo::binarization::internal::singh_formula< V >

Definition at line 58 of file singh_formula.hh.

8.12.2 Member Function Documentation

8.12.2.1 template<typename V > double scribo::binarization::internal::singh_formula< V >::operator() (const V & v, const double *m_x_y*, const double *K*) const

Compute a threshold using Singh's formula.

Parameters

in	<i>v</i>	The current image value.
in	<i>m_x_y</i>	Mean value.
in	<i>K</i>	Control the threshold value in the local window. The higher, the lower the threshold form the local mean $m(x, y)$.

Returns

A threshold.

8.12.2.2 template<typename V > double scribo::binarization::internal::singh_formula< V >::operator() (const V & v, const double *m_x_y*) const

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. $K = 0.34$.

8.13 scribo::binarization::internal::singh_functor< I > Struct Template Reference

Public Types

- enum { **step** = 3 }
- typedef I::value V

Public Member Functions

- **singh_functor** (const Image< I > &input, double *K*)
- void **end_of_row** (int)
- void **exec** (double mean, double stddev)
- void **finalize** ()
- void **init** ()

Public Attributes

- scribo::binarization::internal::singh_formula< V > **formula_**
- const I **input**
- double **K_**
- unsigned **next_line3**

- unsigned offset1
- unsigned offset2
- mln::trait::ch_value< I, bool >
::ret output
- const V * pi
- bool * po

8.13.1 Detailed Description

template<typename I>struct scribo::binarization::internal::singh_funcutor< I >

Definition at line 58 of file singh_funcutor.hh.

8.14 scribo::binarization::internal::wolf_formula< V > Struct Template Reference

Public Member Functions

- **double** [operator\(\)](#) (const **double** m_x_y, const **double** s_x_y, const **double** K, const **double** global_max_stddev, const V &global_min) const

8.14.1 Detailed Description

template<typename V>struct scribo::binarization::internal::wolf_formula< V >

Definition at line 58 of file wolf_formula.hh.

8.14.2 Member Function Documentation

8.14.2.1 template<typename V > **double** scribo::binarization::internal::wolf_formula< V >::operator() (const **double** m_x_y, const **double** s_x_y, const **double** K, const **double** global_max_stddev, const V & global_min) const

Compute a threshold using Wolf's formula.

Returns

A threshold.

8.15 scribo::binarization::internal::wolf_funcutor< I > Struct Template Reference

Public Types

- enum { **step** = 1 }
- typedef I::value V

Public Member Functions

- **wolf_funcutor** (const **Image**< I > &input, **double** K, const typename I::value &global_min, **double** global_max_stddev)
- void **end_of_row** (int)
- void **exec** (**double** mean, **double** stddev)

- void **finalize** ()
- void **init** ()

Public Attributes

- [scribo::binarization::internal::wolf_formula](#)
 < V > **formula_**
- **double** **global_max_stddev_**
- V **global_min_**
- const I **input**
- **double** **K_**
- **unsigned** **next_line**
- **mln::trait::ch_value**< I, bool >
 ::ret **output**
- const I::value * **pi**
- **bool** * **po**

8.15.1 Detailed Description

template<typename I>struct scribo::binarization::internal::wolf_functor< I >

Definition at line 58 of file wolf_functor.hh.

8.16 scribo::binarization::internal::wolf_functor_fast< I > Struct Template Reference

Public Types

- enum { **step** = 3 }
- typedef I::value V

Public Member Functions

- **wolf_functor_fast** (const **Image**< I > &input, **double** K, const typename I::value &global_min, **double** global_max_stddev)
- void **end_of_row** (int)
- void **exec** (**double** mean, **double** stddev)
- void **finalize** ()
- void **init** ()

Public Attributes

- [scribo::binarization::internal::wolf_formula](#)
 < V > **formula_**
- **double** **global_max_stddev_**
- V **global_min_**
- const I **input**
- **double** **K_**
- **unsigned** **next_line3**
- **unsigned** **offset1**
- **unsigned** **offset2**
- **mln::trait::ch_value**< I, bool >
 ::ret **output**

- `const l::value * pi`
- `bool * po`

8.16.1 Detailed Description

template<typename l>struct scribo::binarization::internal::wolf_functor_fast< l >

Definition at line 58 of file wolf_functor_fast.hh.

8.17 scribo::component_features_data Struct Reference

Public Attributes

- `float boldness`
- `scribo::def::color_type color`
- `bool valid`

8.17.1 Detailed Description

Definition at line 39 of file component_features_data.hh.

8.18 scribo::component_info< L > Class Template Reference

```
#include <scribo/core/component_info.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef [component_info](#)< L > **exact_t**

Public Member Functions

- **component_info** (const **component_id_t** &id, const **mln::box2d** &bbox, const **mln::point2d** &mass_center, **unsigned** card, [component::Type](#) type=[component::Undefined](#))
- void **accept** (const [SerializeVisitor](#)< E2 > &visitor) const
- const **mln::box2d** & **bbox** () const
- **unsigned** **card** () const
- const [component_features_data](#) & **features** () const
- **bool** **has_features** () const
- **component_id_t** **id** () const
- **bool** **is_valid** () const
- const **mln::point2d** & **mass_center** () const
- [component::Tag](#) **tag** () const
- [component::Type](#) **type** () const
- void **update_features** (const [component_features_data](#) &features)
- void **update_tag** ([component::Tag](#) tag)
- void **update_type** ([component::Type](#) type)

Protected Attributes

- `mln::box2d` `bbox_`
- `unsigned` `card_`
- `component_features_data` `features_`
- `component_id_t` `id_`
- `mln::point2d` `mass_center_`
- `component::Tag` `tag_`
- `component::Type` `type_`

8.18.1 Detailed Description

```
template<typename L>class scribo::component_info< L >
```

Component information data structure.

Definition at line 64 of file `component_info.hh`.

8.18.2 Member Function Documentation

8.18.2.1 `void scribo::Serializable< component_info< L > >::accept (const SerializeVisitor< E2 > & visitor)`
`const` `[inherited]`

Allow this object to be serialized by `visitor`.

8.19 scribo::component_set< L > Class Template Reference

```
#include <scribo/core/component_set.hh>
```

Public Types

- typedef `Object< void >` `category`
- typedef `component_set< L >` `exact_t`

Public Member Functions

- `component_set` ()
- `component_set` (const `mln::util::tracked_ptr< data_t >` &`data`)
- `component_set` (const `L` &`ima`, const typename `L::value` &`ncomps`)
- `component_set` (const `L` &`ima`, const typename `L::value` &`ncomps`, const `mln::util::array< pair_accu_t >` &`attribs`, `component::Type` `type=component::Undefined`)
- `component_set` (const `L` &`ima`, const typename `L::value` &`ncomps`, const `mln::util::array< pair_data_t >` &`attribs`, `component::Type` `type=component::Undefined`)
- void `accept` (const `SerializeVisitor< E2 >` &`visitor`) const
- void `add_separators` (const typename `mln::trait::ch_value< L, bool >::ret` &`ima`)
- void `clear_separators` ()
- `component_set< L >` `duplicate` () const
- `bool` `has_separators` () const
- const `component_info< L >` &`info` (const typename `L::value` &`id`) const
- `component_info< L >` &`info` (const typename `L::value` &`id`)
- `bool` `is_valid` () const
- const `L` &`labeled_image` () const

- **L::value nelements** () const
- **component_info**< L > & **operator**() (const **component_id_t** &id)
- const **component_info**< L > & **operator**() (const **component_id_t** &id) const
- const **mln::trait::ch_value**< L, **bool** >::ret & **separators** () const
- template<typename F >
void **update_tags** (const **mln::Function_v2b**< F > &f, **component::Tag** tag)

Protected Member Functions

- void **init_** (const **component_set**< L > &model)

Protected Attributes

- **mln::util::tracked_ptr**< **data_t** > **data_**

Related Functions

(Note that these are not member functions.)

- template<typename L >
std::ostream & **operator**<< (std::ostream &ostr, const **component_set**< L > &info)
- template<typename L >
bool operator== (const **component_set**< L > &lhs, const **component_set**< L > &rhs)

8.19.1 Detailed Description

template<typename L>class scribo::component_set< L >

Represents all the components in a document image.

This structure is used to store rich information related to components in an image.

It can be constructed directly from a labeled image and the number of components, or using **primitive::extract::components** (easier and recommended) with a binary image.

```
mln::image2d<bool> ima;
mln::io::pbm::load(ima, "document.pbm");

typedef scribo::def::lbl_type V;
typedef mln::image2d<V> L;
V ncomps;
component_set<L> comps = primitive::extract::components
    (ima, c4(), ncomps);
```

Each component is associated to an id, **component_id_t**. This id is equivalent to the component label in the label image (**labeled_image()**). They are labeled from 1 to **nelements()** included. Id 0 is reserved for the background which is not considered as an actual component.

A **component_set** is considered as valid if it has been initialized (i.e. not instantiated with the default constructor).

On construction, information is automatically computed for each components. To iterate over all the components and get that information, use **operator>()** or **info()**:

```
for_all_comps(c, comps)
{
    const scribo::component_info<L>& comp_info = comps.
        info(c);

    // Some code here...
}
```

Among component information, a [component::Tag](#) tag is stored and used to mark specific components. This tag can be massively updated for all components using [update_tags\(\)](#). The function passed to this method must implement the following interface:

```
template <typename L>
struct my_function
: mln::Function_v2b< my_function<L> >
{
    // Constructor
    my_function(const scribo::component_set<L>&
                components)
        : components_(components)
    {
    }

    // Core function
    bool operator()(const mln_value(L)& l) const
    {
        if (l == mln::literal::zero)
            return false;
        return true; // your test here
    }

    scribo::component_set<L> components_;
};
```

Tags can be used to filter/make a selection of components. Routines performing this selection/filtering automatically are listed in [Component Filtering](#) section.

Components considered as separators in the input image (lines, blocks, ...) may be processed separately and are useful of the rest of the layout analysis. This structure allows you to store a binary image of separators which will be used in some routines to prevent wrong regrouping. Note that no component information will be computed for separators. The separator image can be managed with [has_separators\(\)](#), [add_separators\(\)](#), [separators\(\)](#), and [clear_separators\(\)](#).

[scribo::component_set](#) is the first data structure in the hierarchical representation of a document as explained in [Data structures](#).

See Also

[component_info](#), [component::Tag](#), [component::Type](#), [Data structures](#), [Component Filtering](#).

Definition at line 217 of file `component_set.hh`.

8.19.2 Constructor & Destructor Documentation

8.19.2.1 `template<typename L> scribo::component_set<L>::component_set()`

Default.

8.19.2.2 `template<typename L> scribo::component_set<L>::component_set(const mln::util::tracked_ptr<data_t> & data)`

Constructor from internal data.

8.19.2.3 `template<typename L> scribo::component_set<L>::component_set(const L & ima, const typename L::value & ncomps)`

Constructor from an image `ima` and the number of labels `ncomps`.

8.19.2.4 `template<typename L> scribo::component_set< L >::component_set (const L & ima, const typename L::value & ncomps, const mln::util::array< pair_accu_t > & attrs, component::Type type = component::Undefined)`

Constructor from an image *ima*, the number of labels *ncomps* and attributes values (bounding box and mass center).

8.19.2.5 `template<typename L> scribo::component_set< L >::component_set (const L & ima, const typename L::value & ncomps, const mln::util::array< pair_data_t > & attrs, component::Type type = component::Undefined)`

Constructor from an image *ima*, the number of labels *ncomps* and attributes values (bounding box and mass center).

8.19.3 Member Function Documentation

8.19.3.1 `void scribo::Serializable< component_set< L > >::accept (const SerializeVisitor< E2 > & visitor) const [inherited]`

Allow this object to be serialized by *visitor*.

8.19.3.2 `template<typename L> void scribo::component_set< L >::add_separators (const typename mln::trait::ch_value< L, bool >::ret & ima)`

Add a new separator binary image or perform a logical OR with the existing one.

8.19.3.3 `template<typename L> void scribo::component_set< L >::clear_separators ()`

Clear separator image.

8.19.3.4 `template<typename L> component_set<L> scribo::component_set< L >::duplicate () const`

Create a copy of this `component_set<L>`

8.19.3.5 `template<typename L> bool scribo::component_set< L >::has_separators () const`

Return true if an image of separator exists.

8.19.3.6 `template<typename L> const component_info<L>& scribo::component_set< L >::info (const typename L::value & id) const`

Return component information for a given component id *id*.

8.19.3.7 `template<typename L> component_info<L>& scribo::component_set< L >::info (const typename L::value & id)`

Return component information for a given component id *id*.

8.19.3.8 `template<typename L> void scribo::component_set< L >::init_ (const component_set< L > & model)`
`[protected]`

INTERNAL_API Duplicate the underlying image and create a new [component_set](#).

8.19.3.9 `template<typename L> bool scribo::component_set< L >::is_valid () const`

A [component_set](#) is considered as valid if it has been initialized (i.e. not instantiated with the default constructor).

8.19.3.10 `template<typename L> const L& scribo::component_set< L >::labeled_image () const`

Return the underlying labeled image.

8.19.3.11 `template<typename L> L::value scribo::component_set< L >::nelements () const`

Return the number of components, background excluded.

8.19.3.12 `template<typename L> component_info<L>& scribo::component_set< L >::operator() (const component_id_t & id)`

Return component information for a given component id `id`.

8.19.3.13 `template<typename L> const component_info<L>& scribo::component_set< L >::operator() (const component_id_t & id) const`

Return component information for a given component id `id`.

8.19.3.14 `template<typename L> const mln::trait::ch_value< L , bool >::ret& scribo::component_set< L >::separators () const`

Return the binary image of separators.

8.19.3.15 `template<typename L> template<typename F> void scribo::component_set< L >::update_tags (const mln::Function_v2b< F > & f, component::Tag tag)`

Update component tag for which `f` returns 'False', with `tag`.

8.19.4 Friends And Related Function Documentation

8.19.4.1 `template<typename L> std::ostream & operator<< (std::ostream & ostr, const component_set< L > & info)`
`[related]`

8.19.4.2 `template<typename L> bool operator== (const component_set< L > & lhs, const component_set< L > & rhs)` `[related]`

8.20 scribo::debug::arg_data Struct Reference

Public Attributes

- `const char * desc`

- const char * **name**

8.20.1 Detailed Description

Definition at line 39 of file option_parser.hh.

8.21 scribo::debug::internal::logger_ Class Reference

Public Member Functions

- template<typename V >
 [logger_](#) & [operator](#)<< (const V &v)
- [logger_](#) & [operator](#)<< (std::ostream &(*f)(std::ostream &))
- **bool** [is_verbose](#) () const
- **bool** [is_at_verbose_mode](#) ([VerboseMode](#) mode) const
- **bool** [set_default_verbose_mode](#) ([VerboseMode](#) mode)
- [VerboseMode](#) [default_verbose_mode](#) () const
- **bool** [set_verbose_mode](#) ([VerboseMode](#) mode)
- [VerboseMode](#) [verbose_mode](#) () const
- void [set_verbose_prefix](#) (const std::string &prefix)
- void [log](#) ([VerboseMode](#) mode, const std::string &text)
- **bool** [is_enabled](#) () const
- **bool** [is_at_level](#) ([Level](#) level) const
- void [set_level](#) ([Level](#) level)
- [Level](#) [level](#) () const
- void [set_filename_prefix](#) (const char *name)
- const char * [filename_prefix](#) () const
- template<typename I >
 void [log_image](#) ([Level](#) dbg_level, const [Image](#)< I > &ima, const char *name)
- void [start_time_logging](#) ()
- void [stop_time_logging](#) (const std::string &time_title)

Static Public Member Functions

- static [logger_](#) & [instance](#) ()

8.21.1 Detailed Description

Definition at line 99 of file logger.hh.

8.21.2 Member Function Documentation

8.21.2.1 [VerboseMode](#) scribo::debug::internal::logger_::default_verbose_mode () const

Text Logging.

8.21.2.2 `const char* scribo::debug::internal::logger::filename_prefix () const`

Image Logging.

8.21.2.3 `bool scribo::debug::internal::logger::is_at_level (Level level) const`

Image Logging.

8.21.2.4 `bool scribo::debug::internal::logger::is_at_verbose_mode (VerboseMode mode) const`

Text Logging.

8.21.2.5 `bool scribo::debug::internal::logger::is_enabled () const`

Image Logging.

8.21.2.6 `bool scribo::debug::internal::logger::is_verbose () const`

Text Logging.

8.21.2.7 `Level scribo::debug::internal::logger::level () const`

Image Logging.

8.21.2.8 `void scribo::debug::internal::logger::log (VerboseMode mode, const std::string & text)`

Text Logging.

8.21.2.9 `template<typename I> void scribo::debug::internal::logger::log_image (Level dbg_level, const Image<I> & ima, const char * name)`

Image Logging.

8.21.2.10 `template<typename V> logger_& scribo::debug::internal::logger::operator<< (const V & v)`

Quickly logs text. It uses the default log mode to know if the text.

8.21.2.11 `bool scribo::debug::internal::logger::set_default_verbose_mode (VerboseMode mode)`

The default verbose mode used while logging with [operator<<](#).

8.21.2.12 `void scribo::debug::internal::logger::set_filename_prefix (const char * name)`

Image Logging.

8.21.2.13 `void scribo::debug::internal::logger::set_level (Level level)`

Image Logging.

8.21.2.14 **bool** scribo::debug::internal::logger::set_verbose_mode (*VerboseMode mode*)

Set the current verbose mode, filtering the debug output logged through this object.

8.21.2.15 **void** scribo::debug::internal::logger::set_verbose_prefix (*const std::string & prefix*)

Text Logging.

8.21.2.16 **void** scribo::debug::internal::logger::start_time_logging ()

Time Logging.

This class provides timers in order to performs benchmarks inside a program.

Stopping time logging will output the computed time if Verbose mode is higher or equal to scribo::debug::Time.

8.21.2.17 **void** scribo::debug::internal::logger::stop_time_logging (*const std::string & time.title*)

Time Logging.

This class provides timers in order to performs benchmarks inside a program.

Stopping time logging will output the computed time if Verbose mode is higher or equal to scribo::debug::Time.

8.21.2.18 **VerboseMode** scribo::debug::internal::logger::verbose_mode () **const**

Text Logging.

8.22 scribo::debug::opt_data Struct Reference

Public Attributes

- **const char *** **by_default**
- **const char *** **desc**
- **bool**(* **fcheck**)(const std::vector< const char * > &)
- **const char *** **format**
- **int** **n_args**
- **const char *** **name**

8.22.1 Detailed Description

Definition at line 54 of file option_parser.hh.

8.23 scribo::debug::option_parser Class Reference

Public Member Functions

- **option_parser** (const [arg_data](#) arg_desc[], const [toggle_data](#) toggle_desc[], const [opt_data](#) opt_desc[])
- **const char *** **arg** (const char *) **const**
- **bool** **is_enabled** (const char *toggle_name) **const**
- **bool** **is_set** (const char *opt_name) **const**
- **std::string** **opt_value** (const char *opt_name) **const**

- `std::vector< const char * > opt_values` (`const char *opt_name`) `const`
- `bool parse` (`int argc`, `char *argv[]`)
- `void print_help` () `const`
- `void set_verbose_enabled` (`bool b`)

8.23.1 Detailed Description

Definition at line 65 of file `option_parser.hh`.

8.24 scribo::debug::toggle_data Struct Reference

Public Attributes

- `const char * desc`
- `bool enabled`
- `const char * name`

8.24.1 Detailed Description

Definition at line 46 of file `option_parser.hh`.

8.25 scribo::DMax_Functor< E > Class Template Reference

```
#include <scribo/core/concept/dmax_functor.hh>
```

Public Types

- `typedef Object< void > category`
- `typedef E exact_t`

8.25.1 Detailed Description

```
template<typename E>class scribo::DMax_Functor< E >
```

Dmax functor concept.

Definition at line 40 of file `dmax_functor.hh`.

8.26 scribo::doc_serializer< E > Class Template Reference

```
#include <scribo/core/internal/doc_serializer.hh>
```

Public Types

- `typedef Object< void > category`
- `typedef E exact_t`

Public Member Functions

- `template<typename L >`
`void visit (const document< L > &doc) const`
- `template<typename L >`
`void visit (const line_links< L > &llinks) const`
- `template<typename L >`
`void visit (const object_groups< L > &groups) const`
- `template<typename L >`
`void visit (const object_links< L > &links) const`
- `template<typename L >`
`void visit (const component_set< L > &comp_set) const`
- `template<typename L >`
`void visit (const component_info< L > &info) const`
- `template<typename L >`
`void visit (const paragraph_set< L > &parset) const`
- `template<typename L >`
`void visit (const line_info< L > &line) const`

8.26.1 Detailed Description

`template<typename E>class scribo::doc_serializer< E >`

Link functor concept.

Definition at line 49 of file doc_serializer.hh.

8.27 scribo::document< L > Class Template Reference

```
#include <scribo/core/document.hh>
```

Public Types

- `typedef Object< void > category`
- `typedef document< L > exact_t`

Public Member Functions

- `document (const char *filename)`
- `document (const char *filename, const mln::image2d< mln::value::rgb8 > &input)`
- `void accept (const SerializeVisitor< E2 > &visitor) const`
- `const mln::image2d< bool > & binary_image () const`
- `const mln::image2d< bool > & binary_image_wo_seps () const`
- `const component_set< L > & elements () const`
- `const char * filename () const`
- `bool has_elements () const`
- `bool has_hline_seps () const`
- `bool has_text () const`
- `bool has_vline_seps () const`
- `bool has_whitespace_seps () const`
- `mln::def::coord height () const`
- `const mln::image2d< bool > & hline_seps () const`
- `const component_set< L > & hline_seps_comps () const`

- `const mln::image2d< value::rgb8 > & image () const`
- `bool is_open () const`
- `bool is_valid () const`
- `const line_set< L > & lines () const`
- `void open ()`
- `const paragraph_set< L > & paragraphs () const`
- `void set_binary_image (const mln::image2d< bool > &binary_image)`
- `void set_binary_image_wo_seps (const mln::image2d< bool > &binary_image_wo_seps)`
- `void set_elements (const component_set< L > &elements)`
- `void set_filename (const char *name)`
- `void set_hline_separators (const image2d< bool > &line_seps)`
- `void set_hline_separators (const image2d< bool > &line_seps, const component_set< L > &hline_seps_comps)`
- `void set_image (const mln::image2d< value::rgb8 > &image)`
- `void set_paragraphs (const paragraph_set< L > &parset)`
- `void set_vline_separators (const image2d< bool > &vline_seps)`
- `void set_vline_separators (const image2d< bool > &vline_seps, const component_set< L > &vline_seps_comps)`
- `void set_whitespace_separators (const image2d< bool > &whitespace_seps, const component_set< L > &whitespace_seps_comps)`
- `const mln::image2d< bool > & vline_seps () const`
- `const component_set< L > & vline_seps_comps () const`
- `const mln::image2d< bool > & whitespace_seps () const`
- `const component_set< L > & whitespace_seps_comps () const`
- `mln::def::coord width () const`

8.27.1 Detailed Description

`template<typename L>class scribo::document< L >`

Represent document data and structure.

Definition at line 90 of file document.hh.

8.27.2 Member Function Documentation

8.27.2.1 `void scribo::Serializable< document< L > >::accept (const SerializeVisitor< E2 > & visitor) const`
[inherited]

Allow this object to be serialized by `visitor`.

8.27.2.2 `template<typename L> bool scribo::document< L >::has_text () const`

Check whether this document contains text.

If it returns true, that document contains paragraphs, lines and text components.

8.28 scribo::fun::v2b::components_large_filter< L > Struct Template Reference

`#include <scribo/fun/v2b/objects_large_filter.hh>`

Public Types

- typedef **accu::math::count**
< typename L::psite > **card_t**
- typedef **Function_v2b**< void > **category**
- typedef
[components_large_filter](#)< L > **exact_t**
- typedef void **mutable_result**
- typedef **bool** **result**

Public Member Functions

- [components_large_filter](#) (const [component_set](#)< L > &components, **unsigned** max_size)
- **bool** [operator\(\)](#) (const typename **L::value** &l) const

Public Attributes

- const [component_set](#)< L > [components_](#)
- **mln::util::array**< **bool** > [marked_](#)
- **unsigned** [max_size_](#)
- **L::value** [nlabels_](#)

8.28.1 Detailed Description

template<typename L>struct scribo::fun::v2b::components_large_filter< L >

Filter Functor.

Return false for all components which are too large.

Definition at line 62 of file objects_large_filter.hh.

8.28.2 Constructor & Destructor Documentation

8.28.2.1 template<typename L > scribo::fun::v2b::components_large_filter< L >::components_large_filter (const [component_set](#)< L > & components, unsigned *max_size*)

Constructor.

Parameters

in	<i>components</i>	Component bounding boxes.
in	<i>max_size</i>	Maximum component size.

8.28.3 Member Function Documentation

8.28.3.1 template<typename L > **bool** scribo::fun::v2b::components_large_filter< L >::operator() (const typename **L::value** & l) const

Check if the component is large enough.

Parameters

/	A label.
---	----------

Returns

false if the component area is strictly inferior to `max_size_`.

8.28.4 Member Data Documentation

8.28.4.1 `template<typename L> const component_set<L> scribo::fun::v2b::components_large_filter< L >::components_`

The component set to filter.

Definition at line 89 of file `objects_large_filter.hh`.

8.28.4.2 `template<typename L> mln::util::array<bool> scribo::fun::v2b::components_large_filter< L >::marked_ [mutable]`

Has already been taken into account.

Definition at line 95 of file `objects_large_filter.hh`.

8.28.4.3 `template<typename L> unsigned scribo::fun::v2b::components_large_filter< L >::max_size_`

The minimum area.

Definition at line 86 of file `objects_large_filter.hh`.

8.28.4.4 `template<typename L> L::value scribo::fun::v2b::components_large_filter< L >::nlabels_ [mutable]`

The number of labels remaining after filtering.

Definition at line 92 of file `objects_large_filter.hh`.

8.29 scribo::fun::v2b::components_on_border_filter< L > Struct Template Reference

```
#include <scribo/fun/v2b/objects_on_border_filter.hh>
```

Public Types

- typedef `accu::math::count`
 < typename L::psite > `card_t`
- typedef `Function_v2b`< void > `category`
- typedef
 `components_on_border_filter`< L > `exact_t`
- typedef void `mutable_result`
- typedef `bool` `result`

Public Member Functions

- `components_on_border_filter` (const `component_set`< L > &components)
- `bool operator()` (const typename L::value &l) const

Public Attributes

- const **box2d** & **b_**
- const **component_set**< L > **components_**

8.29.1 Detailed Description

template<typename L> struct scribo::fun::v2b::components_on_border_filter< L >

Filter Functor.

Return false for all components which are too large.

Definition at line 61 of file objects_on_border_filter.hh.

8.29.2 Constructor & Destructor Documentation

8.29.2.1 template<typename L > scribo::fun::v2b::components_on_border_filter< L
>::components_on_border_filter (const component_set< L > & components)

Constructor.

Parameters

in	components	Component bounding boxes.
----	------------	---------------------------

8.29.3 Member Function Documentation

8.29.3.1 template<typename L > bool scribo::fun::v2b::components_on_border_filter< L >::operator() (const
typename L::value & l) const

Check if the component is large enough.

Parameters

in	/	A label.
----	---	----------

Returns

false if the component area is strictly inferior to `max_size_`.

8.29.4 Member Data Documentation

8.29.4.1 template<typename L > const box2d& scribo::fun::v2b::components_on_border_filter< L >::b_

Labeled image bounding box.

Definition at line 86 of file objects_on_border_filter.hh.

8.29.4.2 template<typename L > const component_set<L> scribo::fun::v2b::components_on_border_filter< L
>::components_

The component set to filter.

Definition at line 83 of file objects_on_border_filter.hh.

8.30 scribo::fun::v2b::components_small_filter< L > Struct Template Reference

```
#include <scribo/fun/v2b/objects_small_filter.hh>
```

Public Types

- typedef **accu::math::count**
< typename L::psite > **card_t**
- typedef **Function_v2b**< void > **category**
- typedef
[components_small_filter](#)< L > **exact_t**
- typedef void **mutable_result**
- typedef **bool** **result**

Public Member Functions

- [components_small_filter](#) (const [component_set](#)< L > &components, **unsigned** min_size)
- **bool** [operator\(\)](#) (const typename **L::value** &l) const

Public Attributes

- const [component_set](#)< L > [components_](#)
- **mln::util::array**< **bool** > [marked_](#)
- **unsigned** [min_size_](#)
- **L::value** [nlabels_](#)

8.30.1 Detailed Description

```
template<typename L>struct scribo::fun::v2b::components_small_filter< L >
```

Filter Functor.

Return false for all components which are too small.

Definition at line 62 of file [objects_small_filter.hh](#).

8.30.2 Constructor & Destructor Documentation

8.30.2.1 `template<typename L> scribo::fun::v2b::components_small_filter< L >::components_small_filter (const component_set< L > & components, unsigned min_size)`

Constructor.

Parameters

in	<i>components</i>	Component bounding boxes.
in	<i>min_size</i>	Minimum component size.

8.30.3 Member Function Documentation

8.30.3.1 `template<typename L > bool scribo::fun::v2b::components_small_filter< L >::operator() (const typename L::value & /) const`

Check if the component is large enough.

Parameters

/	A label.
---	----------

Returns

false if the component area is strictly inferior to `min_size_`.

8.30.4 Member Data Documentation

8.30.4.1 `template<typename L > const component_set<L> scribo::fun::v2b::components_small_filter< L >::components_`

The component set to filter.

Definition at line 89 of file `objects_small_filter.hh`.

8.30.4.2 `template<typename L > mIn::util::array<bool> scribo::fun::v2b::components_small_filter< L >::marked_ [mutable]`

Has already been taken into account.

Definition at line 95 of file `objects_small_filter.hh`.

8.30.4.3 `template<typename L > unsigned scribo::fun::v2b::components_small_filter< L >::min_size_`

The minimum area.

Definition at line 86 of file `objects_small_filter.hh`.

8.30.4.4 `template<typename L > L::value scribo::fun::v2b::components_small_filter< L >::nlabels_ [mutable]`

The number of labels remaining after filtering.

Definition at line 92 of file `objects_small_filter.hh`.

8.31 scribo::fun::v2b::label_to_bool< L > Struct Template Reference

Public Types

- typedef **Function_v2b**< void > **category**
- typedef `label_to_bool`< L > **exact_t**
- typedef void **mutable_result**
- typedef **bool** **result**

Public Member Functions

- **label_to_bool** (const `mIn::fun::i2v::array`< **bool** > &f)
- **bool operator()** (const L &v) const

Public Attributes

- `mln::fun::i2v::array< bool > f_`

8.31.1 Detailed Description

template<typename L>struct scribo::fun::v2b::label_to_bool< L >

Definition at line 43 of file label_to_bool.hh.

8.32 scribo::fun::v2v::highlight< R > Struct Template Reference

Public Types

- typedef `Function_v2v< void > category`
- typedef `highlight< R > exact_t`
- typedef void `mutable_result`
- typedef `R result`

Public Member Functions

- result `operator()` (const result &v) const

8.32.1 Detailed Description

template<typename R>struct scribo::fun::v2v::highlight< R >

Definition at line 43 of file highlight.hh.

8.33 scribo::group_info Class Reference

Public Member Functions

- `group_info` (unsigned id, const mln::util::array< `component_id_t` > &comps, unsigned pixel_area, const `box2d` &bbox)
- `group_info` (unsigned id, unsigned pixel_area, const `box2d` &bbox, bool valid=false)
- const `box2d` & `bbox` () const
- unsigned `card` () const
- const mln::util::array
< `component_id_t` > & `component_ids` () const
- mln::util::array
< `component_id_t` > & `component_ids_` ()
- unsigned `id` () const
- void `invalidate` ()
- bool `is_valid` () const
- void `merge` (`group_info` &rhs)
- unsigned `pixel_area` () const

8.33.1 Detailed Description

Definition at line 49 of file group_info.hh.

8.34 scribo::internal::component_set_data< L > Struct Template Reference

Public Types

- typedef **mln::accu::shape::bbox**
< typename L::site > **bbox_accu_t**
- typedef **mln::accu::center**
< typename L::site > **center_accu_t**
- typedef **center_accu_t::result** **center_t**
- typedef **mln::accu::pair**
< **bbox_accu_t**, **center_accu_t** > **pair_accu_t**
- typedef std::pair< mln_box(L),
std::pair< typename L::site,
unsigned > > **pair_data_t**

Public Member Functions

- **component_set_data** (const L &ima, const typename **L::value** &ncomps)
- **component_set_data** (const L &ima, const typename **L::value** &ncomps, const **mln::util::array**< **pair_accu_t** > &attrs, [component::Type](#) type=[component::Undefined](#))
- **component_set_data** (const L &ima, const typename **L::value** &ncomps, const **mln::util::array**< **pair_data_t** > &attrs, [component::Type](#) type=[component::Undefined](#))
- **component_set_data** (const L &ima, const typename **L::value** &ncomps, const **mln::util::array**< [scribo::component_info](#)< L > > &infos)
- void **fill_infos** (const **mln::util::array**< **pair_accu_t** > &attrs, [component::Type](#) type=[component::Undefined](#))
- void **fill_infos** (const **mln::util::array**< **pair_data_t** > &attrs, [component::Type](#) type=[component::Undefined](#))
- void **soft_init** (const typename **L::value** ncomps)

Public Attributes

- L ima_
- **mln::util::array**
< [scribo::component_info](#)< L > > infos_
- **L::value** ncomps_
- **mln::trait::ch_value**< L, bool >
::ret separators_

8.34.1 Detailed Description

template<typename L>struct scribo::internal::component_set_data< L >

Definition at line 82 of file component_set.hh.

8.35 scribo::internal::document_data< L > Struct Template Reference

```
#include <scribo/core/document.hh>
```

Public Member Functions

- **document_data** (const char *filename)
- **document_data** (const char *filename, const **mln::image2d**< **mln::value::rgb8** > &input)

Public Attributes

- `mln::image2d< bool > binary_image_`
- `mln::image2d< bool > binary_image_wo_seps_`
- `component_set< L > elements_`
- `std::string filename_`
- `mln::image2d< bool > hline_seps_`
- `component_set< L > hline_seps_comps_`
- `mln::image2d< mln::value::rgb8 > image_`
- `paragraph_set< L > parset_`
- `mln::image2d< bool > vline_seps_`
- `component_set< L > vline_seps_comps_`
- `mln::image2d< bool > whitespace_seps_`
- `component_set< L > whitespace_seps_comps_`

8.35.1 Detailed Description

`template<typename L> struct scribo::internal::document_data< L >`

Data structure for `scribo::document<L>`.

Definition at line 57 of file `document.hh`.

8.36 scribo::internal::line_info_data< L > Struct Template Reference

```
#include <scribo/core/line_info.hh>
```

Public Member Functions

- `line_info_data` (const `line_set< L >` &holder, const `group_info` &group)
- `line_info_data` (const `line_set< L >` &holder, const `mln::util::array< component_id_t >` &component_ids)

Public Attributes

- `int a_height_`
- `int baseline_`
- `stats< float > baseline_clusters_`
- `mln::box2d bbox_`
- `float boldness_`
- `float boldness_reliability_`
- `unsigned char_space_`
- `unsigned char_width_`
- `mln::value::rgb8 color_`
- `float color_reliability_`
- `mln::util::array< component_id_t > component_ids_`
- `component_set< L > components_`
- `int d_height_`
- `mln::box2d ebbox_`
- `bool hidden_`
- `std::string html_text_`
- `bool indented_`
- `object_links< L > links_`

- `int meanline_`
- `stats< float > meanline_clusters_`
- `float orientation_`
- `unsigned pixel_area_`
- `line::ReadingDirection reading_direction_`
- `float reading_orientation_`
- `bool reverse_video_`
- `line::Tag tag_`
- `std::string text_`
- `float text_confidence_`
- `line::Type type_`
- `unsigned word_space_`
- `unsigned x_height_`

8.36.1 Detailed Description

template<typename L>struct scribo::internal::line_info_data< L >

Data structure for `scribo::line_info<I>`.

Definition at line 81 of file `line_info.hh`.

8.37 scribo::internal::line_links_data< L > Struct Template Reference

```
#include <scribo/core/line_links.hh>
```

Public Member Functions

- `line_links_data` (const `line_set< L >` &lines, `unsigned` size)
- `line_links_data` (const `line_set< L >` &lines, `unsigned` size, `line_id_t` value)

Public Attributes

- `mln::util::array< line_id_t > line_to_link_`
- `line_set< L > lines_`

8.37.1 Detailed Description

template<typename L>struct scribo::internal::line_links_data< L >

Data structure for `scribo::line_links<I>`.

Definition at line 54 of file `line_links.hh`.

8.38 scribo::internal::line_set_data< L > Struct Template Reference

```
#include <scribo/core/line_set.hh>
```

Public Member Functions

- **line_set_data** (const [object_groups](#)< L > &comp_set)
- **line_set_data** (const [mIn::util::array](#)< [scribo::line_info](#)< L > > &infos, const [object_groups](#)< L > &comp_set)

Public Attributes

- [component_set](#)< L > **components_**
- [object_groups](#)< L > **groups_**
- [mIn::util::array](#)< [scribo::line_info](#)< L > > **infos_**
- [object_links](#)< L > **links_**

8.38.1 Detailed Description

template<typename L>struct scribo::internal::line_set_data< L >

Data structure for scribo::line_set<I>.

Definition at line 69 of file line_set.hh.

8.39 scribo::internal::object_groups_data< L > Struct Template Reference

```
#include <scribo/core/object_groups.hh>
```

Public Member Functions

- **object_groups_data** (const [object_links](#)< L > &links)
- **object_groups_data** (const [object_links](#)< L > &links, const [mIn::util::array](#)< [group_info](#) > &info)

Public Attributes

- [mIn::util::array](#)< unsigned > **comp_to_group_**
- [component_set](#)< L > **components_**
- [mIn::util::array](#)< [group_info](#) > **group_info_**
- [object_links](#)< L > **links_**

8.39.1 Detailed Description

template<typename L>struct scribo::internal::object_groups_data< L >

Data structure for scribo::object_groups<I>.

Definition at line 71 of file object_groups.hh.

8.40 scribo::internal::object_links_data< L > Struct Template Reference

```
#include <scribo/core/object_links.hh>
```


Public Member Functions

- **object_links_data** (const [component_set](#)< L > &components, **unsigned** size)
- **object_links_data** (const [component_set](#)< L > &components, **unsigned** size, **unsigned** default_link_id)

Public Attributes

- [mIn::util::array](#)< **unsigned** > **comp_to_link_**
- [component_set](#)< L > **components_**

8.40.1 Detailed Description

template<typename L>struct scribo::internal::object_links_data< L >

Data structure for scribo::object_links<I>.

Definition at line 56 of file object_links.hh.

8.41 scribo::internal::paragraph_set_data< L > Struct Template Reference

```
#include <scribo/core/paragraph_set.hh>
```

Public Member Functions

- **paragraph_set_data** (const [line_links](#)< L > &lines, **unsigned** npars)

Public Attributes

- [line_set](#)< L > **lines_**
- [line_links](#)< L > **links_**
- [mIn::util::array](#)
< [paragraph_info](#)< L > > **pars_**

8.41.1 Detailed Description

template<typename L>struct scribo::internal::paragraph_set_data< L >

Data structure for scribo::paragraph_set<I>.

Definition at line 51 of file paragraph_set.hh.

8.42 scribo::internal::sort_comp_ids< L > Struct Template Reference

Public Member Functions

- **sort_comp_ids** (const [component_set](#)< L > &comp_set)
- **bool operator()** (const [component_id_t](#) &l, const [component_id_t](#) &r) const

Public Attributes

- [component_set](#)< L > **comps_**

8.42.1 Detailed Description

template<typename L>struct scribo::internal::sort_comp_ids< L >

Definition at line 44 of file sort_comp_ids.hh.

8.43 scribo::io::img::internal::debug_img_visitor< L > Class Template Reference

Public Types

- typedef **Object**< void > **category**
- typedef [debug_img_visitor](#)< L > **exact_t**

Public Member Functions

- **debug_img_visitor** (mln::image2d< value::rgb8 > &out, unsigned output_ratio)
- void **visit** (const [line_links](#)< L > &links) const
- void **visit** (const [object_groups](#)< L > &groups) const
- void **visit** (const [object_links](#)< L > &links) const
- void **visit** (const [component_set](#)< L > &comp_set) const
- void **visit** (const [document](#)< L > &doc) const
- void **visit** (const [component_info](#)< L > &info) const
- void **visit** (const [paragraph_set](#)< L > &parset) const
- void **visit** (const [line_info](#)< L > &line) const

8.43.1 Detailed Description

template<typename L>class scribo::io::img::internal::debug_img_visitor< L >

Definition at line 66 of file debug_img_visitor.hh.

8.44 scribo::io::img::internal::full_img_visitor< L > Class Template Reference

Public Types

- typedef **Object**< void > **category**
- typedef [full_img_visitor](#)< L > **exact_t**

Public Member Functions

- **full_img_visitor** (mln::image2d< value::rgb8 > &out)
- void **visit** (const [line_links](#)< L > &links) const
- void **visit** (const [object_groups](#)< L > &groups) const
- void **visit** (const [object_links](#)< L > &links) const
- void **visit** (const [component_set](#)< L > &comp_set) const
- void **visit** (const [document](#)< L > &doc) const
- void **visit** (const [component_info](#)< L > &info) const
- void **visit** (const [paragraph_set](#)< L > &parset) const
- void **visit** (const [line_info](#)< L > &line) const

8.44.1 Detailed Description

template<typename L> class scribo::io::img::internal::full_img_visitor< L >

Definition at line 63 of file full_img_visitor.hh.

8.45 scribo::io::img::internal::non_text_img_visitor Class Reference

```
#include <scribo/io/img/internal/non_text_img_visitor.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef [non_text_img_visitor](#) **exact_t**

Public Member Functions

- **non_text_img_visitor** (mln::image2d< value::rgb8 > &out)
- void **visit** (const [line_links](#)< L > &llinks) const
- void **visit** (const [object_groups](#)< L > &groups) const
- void **visit** (const [object_links](#)< L > &links) const
- template<typename L >
void **visit** (const [document](#)< L > &doc) const
- template<typename L >
void **visit** (const [component_set](#)< L > &comp_set) const
- template<typename L >
void **visit** (const [component_info](#)< L > &info) const
- void **visit** (const [paragraph_set](#)< L > &parset) const
- void **visit** (const [line_info](#)< L > &line) const

8.45.1 Detailed Description

Save non-text information as an image.

Definition at line 57 of file non_text_img_visitor.hh.

8.46 scribo::io::img::internal::text_img_visitor Class Reference

Public Types

- typedef **Object**< void > **category**
- typedef [text_img_visitor](#) **exact_t**

Public Member Functions

- **text_img_visitor** (mln::image2d< value::rgb8 > &)
- void **visit** (const [line_links](#)< L > &llinks) const
- void **visit** (const [object_groups](#)< L > &groups) const
- void **visit** (const [object_links](#)< L > &links) const
- template<typename L >
void **visit** (const [document](#)< L > &doc) const

- void **visit** (const [component_info](#)< L > &[info](#)) const
- template<typename L >
void **visit** (const [component_set](#)< L > &comp_set) const
- template<typename L >
void **visit** (const [paragraph_set](#)< L > &parset) const
- template<typename L >
void **visit** (const [line_info](#)< L > &[line](#)) const

8.46.1 Detailed Description

Definition at line 59 of file text_img_visitor.hh.

8.47 scribo::io::xml::internal::color_t Struct Reference

Public Attributes

- const char * **name**
- float **res**
- const [algebra::vec](#)
< 3, [value::int_u8](#) > **v**

8.47.1 Detailed Description

Definition at line 57 of file compute_text_colour.hh.

8.48 scribo::io::xml::internal::extended_page_xml_visitor< L > Class Template Reference

Public Types

- typedef **Object**< void > **category**
- typedef
[extended_page_xml_visitor](#)< L > **exact_t**

Public Member Functions

- **extended_page_xml_visitor** (std::ofstream &out)
- void **visit** (const [line_links](#)< L > &llinks) const
- void **visit** (const [object_groups](#)< L > &groups) const
- void **visit** (const [object_links](#)< L > &links) const
- void **visit** (const [document](#)< L > &doc) const
- void **visit** (const [component_set](#)< L > &comp_set) const
- void **visit** (const [component_info](#)< L > &[info](#)) const
- void **visit** (const [paragraph_set](#)< L > &parset) const
- void **visit** (const [line_info](#)< L > &[line](#)) const

8.48.1 Detailed Description

template<typename L>class scribo::io::xml::internal::extended_page_xml_visitor< L >

Definition at line 75 of file extended_page_xml_visitor.hh.

8.49 scribo::io::xml::internal::full_xml_visitor Class Reference

Public Types

- typedef **Object**< void > **category**
- typedef [full_xml_visitor](#) **exact_t**

Public Member Functions

- **full_xml_visitor** (std::ofstream &out)
- template<typename L >
void **visit** (const [document](#)< L > &doc) const
- template<typename L >
void **visit** (const [line_links](#)< L > &links) const
- template<typename L >
void **visit** (const [object_groups](#)< L > &groups) const
- template<typename L >
void **visit** (const [object_links](#)< L > &links) const
- template<typename L >
void **visit** (const [component_set](#)< L > &comp_set) const
- template<typename L >
void **visit** (const [component_info](#)< L > &info) const
- template<typename L >
void **visit** (const [paragraph_set](#)< L > &parset) const
- template<typename L >
void **visit** (const [line_info](#)< L > &line) const

8.49.1 Detailed Description

Definition at line 67 of file full_xml_visitor.hh.

8.50 scribo::io::xml::internal::page_xml_visitor< L > Class Template Reference

```
#include <scribo/io/xml/internal/page_xml_visitor.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef [page_xml_visitor](#)< L > **exact_t**

Public Member Functions

- **page_xml_visitor** (std::ofstream &out)
- void **visit** (const [line_links](#)< L > &links) const
- void **visit** (const [object_groups](#)< L > &groups) const
- void **visit** (const [object_links](#)< L > &links) const
- void **visit** (const [line_info](#)< L > &line) const
- void **visit** (const [document](#)< L > &doc) const
- void **visit** (const [component_set](#)< L > &comp_set) const
- void **visit** (const [component_info](#)< L > &info) const
- void **visit** (const [paragraph_set](#)< L > &parset) const

8.50.1 Detailed Description

`template<typename L>class scribo::io::xml::internal::page_xml_visitor< L >`

Save document information as XML.

We use a XML Schema part of the PAGE (Page Analysis and Ground truth Elements) image representation framework.

This schema was used in the Historical Document Layout Analysis COMPetition (HDLAC) for ICDAR 2011.

Its XSD file is located here: <http://schema.primaresearch.org/PAGE/gts/pagecontent/2010-03-19/pagecontent.xsd>

Definition at line 72 of file `page_xml_visitor.hh`.

8.51 scribo::layout::internal::hist_info Struct Reference

Public Attributes

- **unsigned horizontal**
- **unsigned vertical**

8.51.1 Detailed Description

Definition at line 44 of file `hist_info.hh`.

8.52 scribo::layout::internal::node< B > Class Template Reference

Public Member Functions

- **node** (const B &**bbox**)
- const B & **get_bbox** () const
- [node](#)< B > * **get_ls** ()
- const [node](#)< B > * **get_ls** () const
- [node](#)< B > * **get_rs** ()
- const [node](#)< B > * **get_rs** () const
- **bool is_leaf** () const
- void **set_ls** ([node](#)< B > *n)
- void **set_rs** ([node](#)< B > *n)

8.52.1 Detailed Description

`template<typename B>class scribo::layout::internal::node< B >`

Definition at line 47 of file `node.hh`.

8.53 scribo::line_info< L > Class Template Reference

Public Types

- typedef **Object**< void > **category**
- typedef [line_info](#)< L > **exact_t**

Public Member Functions

- **int a_height** () const
- void **accept** (const [SerializeVisitor](#)< E2 > &visitor) const
- **int ascent** () const
- **int baseline** () const
- const **mln::box2d** & **bbox** () const
- **float boldness** () const
- **float boldness_reliability** () const
- **unsigned card** () const
- **unsigned char_space** () const
- **unsigned char_width** () const
- **bool chars_same_width** () const
- const **mln::value::rgb8** & **color** () const
- **float color_reliability** () const
- const **mln::util::array**
 < **component_id_t** > & **component_ids** () const
- **int d_height** () const
- **int delta_of_line** () const
- **int descent** () const
- const **mln::box2d** & **ebbox** () const
- void **force_stats_update** ()
- **unsigned get_first_char_height** () const
- **bool has_text** () const
- const std::string & **html_text** () const
- **line_id_t id** () const
- **bool indented** () const
- **bool is_textline** () const
- **bool is_valid** () const
- **int meanline** () const
- **line_info**< L > & **operator=** (const **line_info**< L > &other)
- **float orientation** () const
- **unsigned pixel_area** () const
- **line::ReadingDirection reading_direction** () const
- **float reading_orientation** () const
- **bool reverse_video** () const
- **line::Tag tag** () const
- const std::string & **text** () const
- **float text_confidence** () const
- **line::Type type** () const
- void **update_ebbox** ()
- void **update_tag** (**line::Tag** tag)
- void **update_text** (const std::string &str, **float** confidence=100.0f)
- void **update_type** (**line::Type** type)
- **unsigned word_space** () const
- **unsigned x_height** () const

- **line_info** ()
- **line_info** (const **line_id_t** &id, **data_t** *data)
- **line_info** (const **line_set**< L > &holder, const **line_id_t** &id, const **group_info** &group)
- **line_info** (const **line_info**< L > &other)

- **bool is_hidden** () const
- void **set_hidden** (**bool** b)

- void **fast_merge** (**line_info**< L > &other, **bool** hide=true)
- void **precise_merge** (**line_info**< L > &other, **bool** hide=true)

8.53.1 Detailed Description

`template<typename L> class scribo::line_info< L >`

Definition at line 160 of file `line_info.hh`.

8.53.2 Constructor & Destructor Documentation

8.53.2.1 `template<typename L> scribo::line_info< L >::line_info ()`

Constructors.

8.53.2.2 `template<typename L> scribo::line_info< L >::line_info (const line_id_t & id, data_t * data)`

Constructors.

8.53.2.3 `template<typename L> scribo::line_info< L >::line_info (const line_set< L > & holder, const line_id_t & id, const group_info & group)`

Constructors.

8.53.2.4 `template<typename L> scribo::line_info< L >::line_info (const line_info< L > & other)`

The line id of the target instance is preserved if it is valid.

8.53.3 Member Function Documentation

8.53.3.1 `void scribo::Serializable< line_info< L > >::accept (const SerializeVisitor< E2 > & visitor) const`
[inherited]

Allow this object to be serialized by `visitor`.

8.53.3.2 `template<typename L> int scribo::line_info< L >::delta_of_line () const`

Returns the delta used to compute the extended bbox.

8.53.3.3 `template<typename L> const mln::box2d& scribo::line_info< L >::ebbox () const`

Extended bounding box.

The width is extended with `char_width()` + `char_space()` on each side. The height is adjusted to `max(a_height, -d_height)` on each side.

8.53.3.4 `template<typename L> void scribo::line_info< L >::fast_merge (line_info< L > & other, bool hide = true)`

Merge related routines.

This merge only updates the component list and the bounding box.

After this merge, the line is tagged with `line::Needs_Precise_Stats_Update`.

The `other` line is tagged with `line::Merged` and if `hide` is set to 'True', it is set as hidden as well.

8.53.3.5 `template<typename L> void scribo::line_info< L >::force_stats_update ()`

Force a new computation of statistics.

8.53.3.6 `template<typename L> line_id_t scribo::line_info< L >::id () const`

If the line info is valid, the line id never changes for a given instance.

8.53.3.7 `template<typename L> bool scribo::line_info< L >::is_hidden () const`

Hidden status.

When a line is hidden, it should not be used in routines computing data over lines.

8.53.3.8 `template<typename L> line_info<L>& scribo::line_info< L >::operator= (const line_info< L > & other)`

The line id of the target instance is preserved if it is valid.

8.53.3.9 `template<typename L> void scribo::line_info< L >::precise_merge (line_info< L > & other, bool hide = true)`

This merge updates the component list and recompute from scratch statistics, bounding box and other line attributes.

After this merge, the line is tagged with line::None.

8.53.3.10 `template<typename L> void scribo::line_info< L >::set_hidden (bool b)`

Hidden status.

When a line is hidden, it should not be used in routines computing data over lines.

8.53.3.11 `template<typename L> void scribo::line_info< L >::update_ebbox ()`

Update the extended bbox.

8.54 scribo::line_links< L > Class Template Reference

```
#include <scribo/core/line_links.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef `line_links`< L > **exact_t**

Public Member Functions

- `line_links` (const `line_set`< L > &lines)
- `line_links` (const `line_set`< L > &lines, `line_id_t` value)
- void `accept` (const `SerializeVisitor`< E2 > &visitor) const
- `line_links`< L > **duplicate** () const
- void **init** ()

- **bool** `is_valid` () const
- const `mln::util::array`
 < `line_id_t` > & `line_to_link` () const
- const `line_set`< L > & `lines` () const
- **unsigned** `nelements` () const
- `line_id_t` & `operator`() (`line_id_t` comp_id)
- const `line_id_t` & `operator`() (`line_id_t` comp_id) const

8.54.1 Detailed Description

template<typename L>class scribo::line_links< L >

Line links representation.

Definition at line 75 of file line_links.hh.

8.54.2 Member Function Documentation

8.54.2.1 `void scribo::Serializable< line_links< L > >::accept (const SerializeVisitor< E2 > & visitor) const`
[inherited]

Allow this object to be serialized by `visitor`.

8.55 scribo::line_set< L > Class Template Reference

```
#include <scribo/core/line_set.hh>
```

Public Member Functions

- const `component_set`< L > & `components` () const
- `component_set`< L > & `components_` ()
- void `compute_lines` (const `object_groups`< L > &groups)
- `line_set`< L > `duplicate` () const
- const `object_groups`< L > & `groups` () const
- const `line_info`< L > & `info` (const typename `L::value` &id) const
- `line_info`< L > & `info` (const typename `L::value` &id)
- const `mln::util::array`
 < `line_info`< L > > & `infos` () const
- **bool** `is_valid` () const
- const `object_links`< L > & `links` () const
- `L::value` `nelements` () const
- `line_info`< L > & `operator`() (const `line_id_t` &id)
- const `line_info`< L > & `operator`() (const `line_id_t` &id) const
- void `update_line_data_` (const `mln::util::array`< `line_info`< L > > &line_data)
- template<typename F >
 void `update_tags` (const `mln::Function_v2b`< F > &f, line::Tag tag)
- template<typename F >
 void `update_types` (const `mln::Function_v2b`< F > &f, line::Type type)
- `line_set` ()
- `line_set` (const `object_groups`< L > &groups)
- `line_set` (const `object_groups`< L > &groups, const `mln::util::array`< `line_info`< L > > &line_data)

- void [force_stats_update](#) ()
- const **mln::util::array**
`< scribo::line_info< L > > & infos_ () const`

8.55.1 Detailed Description

`template<typename L> class scribo::line_set< L >`

Lines container.

Line ids start from 1.

Definition at line 94 of file `line_set.hh`.

8.55.2 Constructor & Destructor Documentation

8.55.2.1 `template<typename L> scribo::line_set< L >::line_set ()`

Constructors

Constructor without argument.

8.55.2.2 `template<typename L> scribo::line_set< L >::line_set (const object_groups< L > & groups)`

Constructor from object groups.

8.55.2.3 `template<typename L> scribo::line_set< L >::line_set (const object_groups< L > & groups, const mln::util::array< line_info< L > > & line_data)`

Constructor useful for delayed construction (loading from file).

8.55.3 Member Function Documentation

8.55.3.1 `template<typename L> const component_set<L> & scribo::line_set< L >::components () const`

Return the underlying component set.

8.55.3.2 `template<typename L> component_set<L> & scribo::line_set< L >::components_ ()`

Return the underlying component set (non-const version).

8.55.3.3 `template<typename L> void scribo::line_set< L >::compute_lines (const object_groups< L > & groups)`

Compute line stats and fill the underlying information.

8.55.3.4 `template<typename L> line_set<L> scribo::line_set< L >::duplicate () const`

Create a copy of this `line_set<L>`

8.55.3.5 `template<typename L> void scribo::line_set< L >::force_stats_update ()`

Massive line computation.

8.55.3.6 `template<typename L> const object_groups<L>& scribo::line_set< L >::groups () const`

Return the underlying component group.

8.55.3.7 `template<typename L> const line_info<L>& scribo::line_set< L >::info (const typename L::value & id) const`

Return line information for a given line id `id`.

8.55.3.8 `template<typename L> line_info<L>& scribo::line_set< L >::info (const typename L::value & id)`

Return line information for a given line id `id`.

8.55.3.9 `template<typename L> const mln::util::array<line_info<L> >& scribo::line_set< L >::infos () const`

Return all the line information.

8.55.3.10 `template<typename L> const mln::util::array<scribo::line_info<L> >& scribo::line_set< L >::infos_() const`

Internal methods.

Return all the line infos.

8.55.3.11 `template<typename L> bool scribo::line_set< L >::is_valid () const`

Return false if it is not initialized (built with the default constructor).

8.55.3.12 `template<typename L> const object_links<L>& scribo::line_set< L >::links () const`

Return the underlying links.

8.55.3.13 `template<typename L> L::value scribo::line_set< L >::nelements () const`

Return the line count.

8.55.3.14 `template<typename L> line_info<L>& scribo::line_set< L >::operator() (const line_id_t & id)`

Return line information for a given line id `id`.

8.55.3.15 `template<typename L> const line_info<L>& scribo::line_set< L >::operator() (const line_id_t & id) const`

Return line information for a given line id `id`.

8.55.3.16 `template<typename L> template<typename F> void scribo::line_set< L >::update_tags (const mln::Function_v2b< F> & f, line::Tag tag)`

Update tag of lines set to 'false' in `f` with `tag`.

8.55.3.17 `template<typename L> template<typename F> void scribo::line_set< L >::update_types (const mIn::Function_v2b< F > & f, line::Type type)`

Update Type of lines set to 'false' in f with tag.

8.56 scribo::Link_Functor< E > Class Template Reference

```
#include <scribo/core/concept/link_functor.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef E **exact_t**

8.56.1 Detailed Description

```
template<typename E>class scribo::Link_Functor< E >
```

Link functor concept.

Definition at line 43 of file link_functor.hh.

8.57 scribo::object_groups< L > Class Template Reference

```
#include <scribo/core/object_groups.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef [object_groups](#)< L > **exact_t**

Public Member Functions

- **object_groups** (const [object_links](#)< L > &links)
- **object_groups** (const [object_links](#)< L > &links, const [mIn::util::array](#)< [group_info](#) > &info)
- void [accept](#) (const [SerializeVisitor](#)< E2 > &visitor) const
- const [mIn::util::array](#) < **unsigned** > & **comp_to_group** () const
- const [component_set](#)< L > & **components** () const
- [object_groups](#)< L > **duplicate** () const
- const [group_info](#) & [group_of](#) (**unsigned** comp_id) const
- [group_info](#) & [group_of](#) (**unsigned** comp_id)
- **bool** **is_valid** () const
- const [object_links](#)< L > & **links** () const
- void **merge** (**unsigned** group_id_from, **unsigned** group_id_to)
- **unsigned** **nelements** () const
- const [group_info](#) & [operator\(\)](#) (**unsigned** group_id) const
- [group_info](#) & [operator\(\)](#) (**unsigned** group_id)

8.57.1 Detailed Description

template<typename L>class scribo::object_groups< L >

Object group representation.

Definition at line 95 of file object_groups.hh.

8.57.2 Member Function Documentation

8.57.2.1 void scribo::Serializable< object_groups< L > >::accept (const SerializeVisitor< E2 > & visitor) const
[inherited]

Allow this object to be serialized by visitor.

8.57.2.2 template<typename L> const group_info& scribo::object_groups< L >::group_of (unsigned comp_id)
const

Return the group id of the component comp_id.

8.57.2.3 template<typename L> const group_info& scribo::object_groups< L >::operator() (unsigned group_id)
const

Return group info data for group with id group_id.

Valid id starts from 1.

8.58 scribo::object_links< L > Class Template Reference

```
#include <scribo/core/object_links.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef [object_links](#)< L > **exact_t**

Public Member Functions

- [object_links](#) ()
- [object_links](#) (const [component_set](#)< L > &components)
- [object_links](#) (const [component_set](#)< L > &components, unsigned default_link_id)
- void [accept](#) (const [SerializeVisitor](#)< E2 > &visitor) const
- const [mIn::util::array](#)
< unsigned > & [comp_to_link](#) () const
- const [component_set](#)< L > & [components](#) () const
- [object_links](#)< L > [duplicate](#) () const
- bool [has_linking_enabled](#) (unsigned comp_id) const
- void [init](#) ()
- bool [is_linked](#) (unsigned comp_id) const
- bool [is_valid](#) () const
- unsigned [nelements](#) () const

- void [update](#) (**unsigned** from_id, **unsigned** to_id)
- void [clear](#) (**unsigned** id)
- void [disable_linking](#) (**unsigned** id)
- const **unsigned** & [operator\(\)](#) (**unsigned** comp_id) const

Related Functions

(Note that these are not member functions.)

- `template<typename L >`
`std::ostream & operator<< (std::ostream &ostr, const object_links< L > &links)`

8.58.1 Detailed Description

`template<typename L>class scribo::object_links< L >`

Object links representation.

This structure is meant to store link information between components. Linking components can be considered as a first step towards component grouping.

It requires a [component_set](#) to be constructed. Each component existing in the [component_set](#) may have link in an `object_link` structure. If no [component_set](#) is used for construction, this object is invalid (

See Also

[is_valid\(\)](#)).

Definition at line 88 of file `object_links.hh`.

8.58.2 Constructor & Destructor Documentation

8.58.2.1 `template<typename L> scribo::object_links< L >::object_links ()`

Default constructor. It produces an invalid structure.

8.58.2.2 `template<typename L> scribo::object_links< L >::object_links (const component_set< L > &components)`

Construct a valid [object_links](#).

Links is enabled for each valid component but no link is set. Invalid components links are disabled.

8.58.2.3 `template<typename L> scribo::object_links< L >::object_links (const component_set< L > &components, unsigned default_link_id)`

Construct a valid [object_links](#).

Links is enabled for each valid component and set by default towards component with id `default_link_id`. Invalid components links are disabled.

8.58.3 Member Function Documentation

8.58.3.1 `void scribo::Serializable< object_links< L > >::accept (const SerializeVisitor< E2 > & visitor) const`
 [inherited]

Allow this object to be serialized by `visitor`.

8.58.3.2 `template<typename L> void scribo::object_links< L >::clear (unsigned id)`

Reset link for component with id `id`.

This component can be linked later.

8.58.3.3 `template<typename L> const mln::util::array<unsigned>& scribo::object_links< L >::comp_to_link () const`

Returns the underlying array encoding the component links.

Indexes in array correspond to component ids and the corresponding value is the component id involved in the link.

8.58.3.4 `template<typename L> const component_set<L>& scribo::object_links< L >::components () const`

Return the underlying [component_set](#).

8.58.3.5 `template<typename L> void scribo::object_links< L >::disable_linking (unsigned id)`

Do not allow component with id `id` to be linked to another component.

8.58.3.6 `template<typename L> object_links<L> scribo::object_links< L >::duplicate () const`

Make a deep copy of this structure.

8.58.3.7 `template<typename L> bool scribo::object_links< L >::has_linking_enabled (unsigned comp_id) const`

Return True if component `comp_id` can be linked to another component.

8.58.3.8 `template<typename L> void scribo::object_links< L >::init ()`

Initialize links.

Each component is linked to itself (i.e. has no link). Invalid components have linking disabled.

8.58.3.9 `template<typename L> bool scribo::object_links< L >::is_linked (unsigned comp_id) const`

Return True if component `comp_id` has a link starting from itself to another component.

8.58.3.10 `template<typename L> bool scribo::object_links< L >::is_valid () const`

Return True if this [object_links](#) structure is correctly constructed.

8.58.3.11 `template<typename L> unsigned scribo::object_links< L >::nelements () const`

Return the number of links.

This is equivalent to the number of components + the background.

8.58.3.12 `template<typename L> const unsigned& scribo::object_links< L >::operator() (unsigned comp_id) const`

Get link id for component `comp_id`.

8.58.3.13 `template<typename L> void scribo::object_links< L >::update (unsigned from_id, unsigned to_id)`

Link related methods.

Set link between component `from_id` and `to_id`.

8.58.4 Friends And Related Function Documentation

8.58.4.1 `template<typename L> std::ostream & operator<< (std::ostream & ostr, const object_links< L > & links)`
[related]

8.59 scribo::paragraph_info< L > Class Template Reference

```
#include <scribo/core/paragraph_info.hh>
```

Public Member Functions

- **paragraph_info** (const [line_links](#)< L > &links)
- void **add_line** (const [line_info](#)< L > &line)
- const **mln::box2d** & **bbox** () const
- const **mln::value::rgb8** & **color** () const
- **float** **color_reliability** () const
- **int** **delta_baseline** () const
- void **fast_merge** ([paragraph_info](#)< L > &info)
- void **force_stats_update** ()
- void **invalidate** ()
- **bool** **is_valid** () const
- const [line_info](#)< L > & **line** (**line_id_t** id) const
- const **mln::util::array**
 < **line_id_t** > & **line_ids** () const
- const [line_links](#)< L > & **llinks** () const
- **bool** **needs_stats_update** () const
- **unsigned** **nlines** () const
- void **set_color_** (const **mln::value::rgb8** &v)
- void **set_color_reliability_** (**float** v)
- void **set_delta_baseline** (const **int** delta_baseline)
- **paragraph::Tag** **tag** () const
- void **update_tag** (**paragraph::Tag** tag)

8.59.1 Detailed Description

```
template<typename L>class scribo::paragraph_info< L >
```

Paragraph structure information.

Definition at line 43 of file `paragraph_info.hh`.

8.59.2 Member Function Documentation

8.59.2.1 `template<typename L> void scribo::paragraph_info< L >::add_line (const line_info< L > & line)`

Add a new line to this paragraph.

This method is provided for an incremental construction.

Once this method has been called, `needs_stats_update()` will return true until `force_stats_update()` is called.

8.60 scribo::paragraph_set< L > Class Template Reference

```
#include <scribo/core/paragraph_set.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef `paragraph_set`< L > **exact_t**

Public Member Functions

- `paragraph_set` (`internal::paragraph_set_data`< L > ***data**)
- `paragraph_set` (const `line_links`< L > &**links**, **unsigned** npars)
- void `accept` (const `SerializeVisitor`< E2 > &**visitor**) const
- `paragraph_set`< L > **duplicate** () const
- template<typename F >
void **invalidate** (const `Function_v2b`< F > &**f**)
- **bool is_valid** () const
- const `line_set`< L > & **lines** () const
- const `line_links`< L > & **links** () const
- **unsigned nelements** () const
- `paragraph_info`< L > & **operator()** (const `paragraph_id_t` &**i**)
- const `paragraph_info`< L > & **operator()** (const `paragraph_id_t` &**i**) const

8.60.1 Detailed Description

```
template<typename L>class scribo::paragraph_set< L >
```

Paragraph container.

Paragraph ids start from 1.

Definition at line 71 of file `paragraph_set.hh`.

8.60.2 Member Function Documentation

8.60.2.1 `void scribo::Serializable< paragraph_set< L > >::accept (const SerializeVisitor< E2 > & visitor) const`
[inherited]

Allow this object to be serialized by `visitor`.

8.61 scribo::preprocessing::internal::Hough Class Reference

Public Member Functions

- **Hough** (int width, int height)
- **image2d**< unsigned > & **acc** ()
- **double get_cos** (int index) const
- **double get_sin** (int index) const
- **int height** () const
- void **look_up_table** ()
- **double mrho** () const
- **int mrhoi** () const
- **double mtheta** () const
- **int mthetai** () const
- **int width** () const

8.61.1 Detailed Description

Definition at line 54 of file deskew.hh.

8.62 scribo::preprocessing::internal::QCompare Struct Reference

Public Member Functions

- **bool operator()** (const [s_angle](#) &s1, const [s_angle](#) &s2)

8.62.1 Detailed Description

Definition at line 102 of file deskew.hh.

8.63 scribo::preprocessing::internal::s_angle Struct Reference

Public Attributes

- **unsigned max**
- **int pos**

8.63.1 Detailed Description

Definition at line 95 of file deskew.hh.

8.64 scribo::primitive::link::internal::dmax_default Class Reference

```
#include <scribo/primitive/link/internal/dmax_default.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef [dmax_default](#) **exact_t**

Public Member Functions

- **dmax_default** (float dmax_factor)
- **float compute_** (const **box2d** &b) const
- **float operator()** (const **box2d** &b) const

Protected Attributes

- **float dmax_factor_**

8.64.1 Detailed Description

Base class for dmax functors.

Definition at line 50 of file dmax_default.hh.

8.65 scribo::primitive::link::internal::dmax_functor_base< E > Class Template Reference

```
#include <scribo/primitive/link/internal/dmax_functor_base.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef E **exact_t**

Public Member Functions

- **dmax_functor_base** (float dmax_factor)
- **float operator()** (const **box2d** &b) const

Protected Attributes

- **float dmax_factor_**

8.65.1 Detailed Description

```
template<typename E>class scribo::primitive::link::internal::dmax_functor_base< E >
```

Base class for dmax functors.

Definition at line 57 of file dmax_functor_base.hh.

8.66 scribo::primitive::link::internal::dmax_hrules Class Reference

```
#include <scribo/primitive/link/internal/dmax_hrules.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef [dmax_hrules](#) **exact_t**

Public Member Functions

- **dmax_hrules** (float dmax_factor, **unsigned** fixed_dmax)
- **float compute_** (const **box2d** &b) const
- **float operator()** (const **box2d** &b) const

Protected Attributes

- **float dmax_factor_**
- **unsigned fixed_dmax_**

8.66.1 Detailed Description

Base class for dmax functors.

Definition at line 50 of file dmax_hrules.hh.

8.67 scribo::primitive::link::internal::dmax_width_and_height Class Reference

```
#include <scribo/primitive/link/internal/dmax_width_and_height.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef [dmax_width_and_height](#) **exact_t**

Public Member Functions

- **dmax_width_and_height** (float dmax_factor)
- **float compute_** (const **box2d** &) const
- **float operator()** (const **box2d** &b) const

Protected Attributes

- **float dmax_factor_**

8.67.1 Detailed Description

Base class for dmax functors.

Definition at line 53 of file dmax_width_and_height.hh.

8.68 scribo::primitive::link::internal::dmax_width_only Class Reference

```
#include <scribo/primitive/link/internal/dmax_width_only.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef **dmax_width_only** **exact_t**

Public Member Functions

- **float compute_** (const **box2d** &) const
- **float operator()** (const **box2d** &b) const

Protected Attributes

- **float dmax_factor_**

8.68.1 Detailed Description

Base class for dmax functors.

Definition at line 53 of file `dmax_width_only.hh`.

8.69 scribo::primitive::link::internal::link_functor_base< L, E > Class Template Reference

```
#include <scribo/primitive/link/internal/link_functor_base.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef **component_set**< L > **component_set_t**
- typedef **mln::util::couple**
 < anchor::Type, P > **couple_t**
- typedef E **exact_t**
- typedef L::site **P**
- typedef L **support**

Public Member Functions

- **link_functor_base** (const **component_set**< L > &components)
- const **component_set**< L > & **components** () const
- void **compute_next_site** (P &p)
- void **compute_next_site_** (P &p)
- **couple_t** **finalize_link** (unsigned current_object)
- **couple_t** **finalize_link_** (unsigned current_object)
- void **initialize_link** (unsigned current_object)
- void **initialize_link_** (unsigned current_object)
- void **invalidate_link** (unsigned current_object, const P &start_point, const P &p, anchor::Type anchor)
- void **invalidate_link** (unsigned current_object, const P &start_point, const P &p)
- void **invalidate_link_** (unsigned current_object, const P &start_point, const P &p, anchor::Type anchor)
- **bool is_potential_link** (unsigned current_object, const P &start_point, const P &p) const
- **bool is_potential_link_** (unsigned current_object, const P &start_point, const P &p) const
- const L & **labeled_image** () const

- **unsigned link** (**unsigned** object) const
- const [object_links](#)< L > & **links** () const
- L::site **start_point** (**unsigned** current_object, anchor::Type anchor)
- L::site [start_point](#) (**unsigned** current_object)
- L::site **start_point_** (**unsigned** current_object, anchor::Type anchor)
- void **start_processing_object** (**unsigned** current_object)
- void **start_processing_object_** (**unsigned** current_object)
- **bool valid_link** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- **bool valid_link_** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- void **validate_link** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- void [validate_link](#) (**unsigned** current_object, const P &start_point, const P &p)
- void **validate_link_** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- **bool verify_link_criterion** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor) const
- **bool verify_link_criterion_** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor) const

Protected Attributes

- const [component_set](#)< L > **components_**
- const L & **labeled_image_**
- [object_links](#)< L > **links_**

8.69.1 Detailed Description

template<typename L, typename E>class scribo::primitive::link::internal::link_funcutor_base< L, E >

Base class for link functors.

Definition at line 70 of file link_funcutor_base.hh.

8.69.2 Constructor & Destructor Documentation

8.69.2.1 template<typename L, typename E> **scribo::primitive::link::internal::link_funcutor_base**< L, E >::link_funcutor_base (const [component_set](#)< L > & *components*)

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts.

8.69.3 Member Function Documentation

8.69.3.1 template<typename L, typename E> void **scribo::primitive::link::internal::link_funcutor_base**< L, E >::invalidate_link (**unsigned** *current_object*, const P & *start_point*, const P & *p*)

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.69.3.2 template<typename L, typename E> L::site **scribo::primitive::link::internal::link_funcutor_base**< L, E >::start_point (**unsigned** *current_object*)

This is an overloaded member function, provided for convenience. **It** differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.69.3.3 `template<typename L, typename E> void scribo::primitive::link::internal::link_functor_base< L, E >::validate_link (unsigned current_object, const P & start_point, const P & p)`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.70 scribo::primitive::link::internal::link_several_dmax_base< L, E > Class Template Reference

```
#include <scribo/primitive/link/internal/link_several_dmax_base.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef `component_set`< L > **component_set_t**
- typedef E **exact_t**
- typedef L **support**

Public Member Functions

- **link_several_dmax_base** (const `component_set`< L > &comps, **unsigned** neighb_max_distance)
- const **util::array**< anchor::Type > & **anchors** () const
- const `component_set`< L > & **components** () const
- void **compute_next_site** (P &p)
- void **compute_next_site_** (P &p)
- **couple_t** **finalize_link** (**unsigned** current_object)
- **couple_t** **finalize_link_** (**unsigned** current_object)
- void **initialize_link** (**unsigned** current_object)
- void **initialize_link_** (**unsigned** current_object)
- void **invalidate_link** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- void **invalidate_link** (**unsigned** current_object, const P &start_point, const P &p)
- void **invalidate_link_** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- **bool** **is_potential_link** (**unsigned** current_object, const P &start_point, const P &p) const
- **bool** **is_potential_link_** (**unsigned** current_object, const P &start_point, const P &p) const
- const L & **labeled_image** () const
- **unsigned** **link** (**unsigned** object) const
- const `object_links`< L > & **links** () const
- **unsigned** **nanchors** () const
- L::site **start_point** (**unsigned** current_object, anchor::Type anchor)
- L::site **start_point** (**unsigned** current_object)
- L::site **start_point_** (**unsigned** current_object, anchor::Type anchor)
- void **start_processing_object** (**unsigned** current_object)
- void **start_processing_object_** (**unsigned** current_object)
- **bool** **valid_link** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- **bool** **valid_link_** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- void **validate_link** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- void **validate_link** (**unsigned** current_object, const P &start_point, const P &p)
- void **validate_link_** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- **bool** **verify_link_criterion** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor) const
- **bool** **verify_link_criterion_** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor) const

Protected Attributes

- `mln::util::array< anchor::Type > anchors_`
- `const component_set< L > components_`
- `anchor::Direction direction_`
- `float dmax_`
- `const L & labeled_image_`
- `object_links< L > links_`
- `float neighb_max_distance_`
- `mln::util::array< couple_t > potential_links_`

8.70.1 Detailed Description

template<typename L, typename E>class scribo::primitive::link::internal::link_several_dmax_base< L, E >

Base class for link functors using several anchors and a maximum lookup distance.

Definition at line 70 of file link_several_dmax_base.hh.

8.70.2 Member Function Documentation

8.70.2.1 template<typename L, typename E> void scribo::primitive::link::internal::link_functor_base< L, E >::invalidate_link (unsigned current_object, const P & start_point, const P & p) [inherited]

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.70.2.2 template<typename L, typename E> L::site scribo::primitive::link::internal::link_functor_base< L, E >::start_point (unsigned current_object) [inherited]

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.70.2.3 template<typename L, typename E> void scribo::primitive::link::internal::link_functor_base< L, E >::validate_link (unsigned current_object, const P & start_point, const P & p) [inherited]

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.71 scribo::primitive::link::internal::link_single_dmax_base< L, E > Class Template Reference

```
#include <scribo/primitive/link/internal/link_single_dmax_base.hh>
```

Public Types

- typedef `Object< void > category`
- typedef `component_set< L > component_set_t`
- typedef `mln::util::couple< anchor::Type, P > couple_t`
- typedef `E exact_t`
- typedef `L::site P`
- typedef `L support`

Public Member Functions

- **link_single_dmax_base** (const [component_set](#)< L > &components, **unsigned** neighb_max_distance, anchor::Direction direction)
- const [component_set](#)< L > & **components** () const
- void **compute_next_site** (P &p)
- void **compute_next_site_** (P &p)
- **couple_t** **finalize_link** (**unsigned** current_object)
- **couple_t** **finalize_link_** (**unsigned** current_object)
- void **initialize_link** (**unsigned** current_object)
- void **initialize_link_** (**unsigned** current_object)
- void **invalidate_link** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- void [invalidate_link](#) (**unsigned** current_object, const P &start_point, const P &p)
- void **invalidate_link_** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- **bool** **is_potential_link** (**unsigned** current_object, const P &start_point, const P &p) const
- **bool** **is_potential_link_** (**unsigned** current_object, const P &start_point, const P &p) const
- const L & **labeled_image** () const
- **unsigned** **link** (**unsigned** object) const
- const [object_links](#)< L > & **links** () const
- L::site **start_point** (**unsigned** current_object, anchor::Type anchor)
- L::site [start_point](#) (**unsigned** current_object)
- L::site **start_point_** (**unsigned** current_object, anchor::Type anchor)
- void **start_processing_object** (**unsigned** current_object)
- void **start_processing_object_** (**unsigned** current_object)
- **bool** **valid_link** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- **bool** **valid_link_** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- void **validate_link** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- void [validate_link](#) (**unsigned** current_object, const P &start_point, const P &p)
- void **validate_link_** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- **bool** **verify_link_criterion** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor) const
- **bool** **verify_link_criterion_** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor) const

Protected Attributes

- const [component_set](#)< L > **components_**
- const L & **labeled_image_**
- [object_links](#)< L > **links_**

8.71.1 Detailed Description

template<typename L, typename E>class scribo::primitive::link::internal::link_single_dmax_base< L, E >

Base class for link functors using mass centers and a given max distance.

Definition at line 69 of file link_single_dmax_base.hh.

8.71.2 Member Function Documentation

8.71.2.1 template<typename L, typename E> void scribo::primitive::link::internal::link_functor_base< L, E >::invalidate_link (**unsigned** *current_object*, const P & *start_point*, const P & *p*) [inherited]

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.71.2.2 `template<typename L, typename E> L::site scribo::primitive::link::internal::link_functor_base< L, E >::start_point (unsigned current_object) [inherited]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.71.2.3 `template<typename L, typename E> void scribo::primitive::link::internal::link_functor_base< L, E >::validate_link (unsigned current_object, const P & start_point, const P & p) [inherited]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.72 scribo::primitive::link::internal::link_single_dmax_ratio_aligned_base< L, F, E > Class Template Reference

Public Types

- typedef **Object**< void > **category**
- typedef **component_set**< L > **component_set_t**
- typedef **mln::util::couple**
< anchor::Type, P > **couple_t**
- typedef E **exact_t**
- typedef L::site **P**
- typedef L **support**

Public Member Functions

- **link_single_dmax_ratio_aligned_base** (const **component_set**< L > &components, const **DMax_Functor**< F > &dmax_f, float min_angle, float max_angle, anchor::Type anchor)
- const **component_set**< L > & **components** () const
- void **compute_next_site** (P &p)
- void **compute_next_site_** (P &p)
- **couple_t** **finalize_link** (unsigned current_object)
- **couple_t** **finalize_link_** (unsigned current_object)
- void **initialize_link** (unsigned current_object)
- void **initialize_link_** (unsigned current_object)
- void **invalidate_link** (unsigned current_object, const P &start_point, const P &p, anchor::Type anchor)
- void **invalidate_link** (unsigned current_object, const P &start_point, const P &p)
- void **invalidate_link_** (unsigned current_object, const P &start_point, const P &p, anchor::Type anchor)
- bool **is_potential_link** (unsigned current_object, const P &start_point, const P &p) const
- bool **is_potential_link_** (unsigned current_object, const P &start_point, const P &p) const
- const L & **labeled_image** () const
- unsigned **link** (unsigned object) const
- const **object_links**< L > & **links** () const
- L::site **start_point** (unsigned current_object, anchor::Type anchor)
- L::site **start_point** (unsigned current_object)
- L::site **start_point_** (unsigned current_object, anchor::Type anchor)
- void **start_processing_object** (unsigned current_object)
- void **start_processing_object_** (unsigned current_object)
- bool **valid_link** (unsigned current_object, const P &start_point, const P &p, anchor::Type anchor)
- bool **valid_link_** (unsigned current_object, const P &start_point, const P &p, anchor::Type anchor)
- void **validate_link** (unsigned current_object, const P &start_point, const P &p, anchor::Type anchor)

- void [validate_link](#) (**unsigned** current_object, const P &start_point, const P &p)
- void [validate_link_](#) (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- **bool** [verify_link_criterion](#) (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor) const
- **bool** [verify_link_criterion_](#) (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor) const

Public Attributes

- **mln::trait::ch_value**< L, **value::rgb8** >::ret [debug_](#)
- anchor::Type [debug_anchor_](#)
- **mln::trait::ch_value**< L, **value::rgb8** >::ret [debug_angle_](#)
- **float** [max_alpha_rad](#)
- **float** [min_alpha_rad](#)

Protected Attributes

- const [component_set](#)< L > [components_](#)
- anchor::Direction [direction_](#)
- **float** [dmax_](#)
- const F [dmax_f_](#)
- const L & [labeled_image_](#)
- [object_links](#)< L > [links_](#)

8.72.1 Detailed Description

template<typename L, typename F, typename E>class scribo::primitive::link::internal::link_single_dmax_ratio_aligned_base< L, F, E >

Definition at line 72 of file link_single_dmax_ratio_aligned_base.hh.

8.72.2 Member Function Documentation

8.72.2.1 template<typename L, typename E> void scribo::primitive::link::internal::link_functor_base< L, E >::invalidate_link (**unsigned** *current_object*, const P & *start_point*, const P & *p*) [inherited]

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.72.2.2 template<typename L, typename E> L::site scribo::primitive::link::internal::link_functor_base< L, E >::start_point (**unsigned** *current_object*) [inherited]

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.72.2.3 template<typename L, typename E> void scribo::primitive::link::internal::link_functor_base< L, E >::validate_link (**unsigned** *current_object*, const P & *start_point*, const P & *p*) [inherited]

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.73 scribo::primitive::link::internal::link_single_dmax_ratio_aligned_delta_base< L, F, E > Class Template Reference

Public Types

- typedef **Object**< void > **category**
- typedef [component_set](#)< L > **component_set_t**
- typedef **mln::util::couple**
< anchor::Type, P > **couple_t**
- typedef E **exact_t**
- typedef L::site **P**
- typedef L **support**

Public Member Functions

- **link_single_dmax_ratio_aligned_delta_base** (const [component_set](#)< L > &components, const [DMax_Functor](#)< F > &dmax_f, **int** delta_pixel, anchor::Direction delta_direction)
- const [component_set](#)< L > & **components** () const
- void **compute_next_site** (P &p)
- void **compute_next_site_** (P &p)
- **couple_t** **finalize_link** (**unsigned** current_object)
- **couple_t** **finalize_link_** (**unsigned** current_object)
- void **initialize_link** (**unsigned** current_object)
- void **initialize_link_** (**unsigned** current_object)
- void **invalidate_link** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- void [invalidate_link](#) (**unsigned** current_object, const P &start_point, const P &p)
- void **invalidate_link_** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- **bool** **is_potential_link** (**unsigned** current_object, const P &start_point, const P &p) const
- **bool** **is_potential_link_** (**unsigned** current_object, const P &start_point, const P &p) const
- const L & **labeled_image** () const
- **unsigned** **link** (**unsigned** object) const
- const [object_links](#)< L > & **links** () const
- L::site **start_point** (**unsigned** current_object, anchor::Type anchor)
- L::site [start_point](#) (**unsigned** current_object)
- L::site **start_point_** (**unsigned** current_object, anchor::Type anchor)
- void **start_processing_object** (**unsigned** current_object)
- void **start_processing_object_** (**unsigned** current_object)
- **bool** **valid_link** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- **bool** **valid_link_** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- void **validate_link** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- void [validate_link](#) (**unsigned** current_object, const P &start_point, const P &p)
- void **validate_link_** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor)
- **bool** **verify_link_criterion** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor) const
- **bool** **verify_link_criterion_** (**unsigned** current_object, const P &start_point, const P &p, anchor::Type anchor) const

Public Attributes

- anchor::Direction **delta_direction_**
- **int** **delta_pixel_**

Protected Attributes

- const [component_set](#)< L > **components_**
- anchor::Direction **direction_**
- float **dmax_**
- const F **dmax_f_**
- const L & **labeled_image_**
- [object_links](#)< L > **links_**

8.73.1 Detailed Description

template<typename L, typename F, typename E>class scribo::primitive::link::internal::link_single_dmax_ratio_aligned_delta_base< L, F, E >

Definition at line 72 of file link_single_dmax_ratio_aligned_delta_base.hh.

8.73.2 Member Function Documentation

8.73.2.1 template<typename L, typename E> void scribo::primitive::link::internal::link_funcutor_base< L, E >::invalidate_link (unsigned *current_object*, const P & *start_point*, const P & *p*) [inherited]

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.73.2.2 template<typename L, typename E> L::site scribo::primitive::link::internal::link_funcutor_base< L, E >::start_point (unsigned *current_object*) [inherited]

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.73.2.3 template<typename L, typename E> void scribo::primitive::link::internal::link_funcutor_base< L, E >::validate_link (unsigned *current_object*, const P & *start_point*, const P & *p*) [inherited]

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.74 scribo::primitive::link::internal::link_single_dmax_ratio_base< L, F, E > Class Template Reference

```
#include <scribo/primitive/link/internal/link_single_dmax_ratio_base.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef [component_set](#)< L > **component_set_t**
- typedef `mln::util::couple`
< anchor::Type, P > **couple_t**
- typedef E **exact_t**
- typedef L::site **P**
- typedef L **support**

Public Member Functions

- **link_single_dmax_ratio_base** (const [component_set](#)< L > &components, anchor::Direction direction, const [DMax_Functor](#)< F > &dmax_f)
- const [component_set](#)< L > &**components** () const
- void **compute_next_site** (P &p)
- void **compute_next_site_** (P &p)
- **couple_t** **finalize_link** (unsigned current_object)
- **couple_t** **finalize_link_** (unsigned current_object)
- void **initialize_link** (unsigned current_object)
- void **initialize_link_** (unsigned current_object)
- void **invalidate_link** (unsigned current_object, const P &start_point, const P &p, anchor::Type anchor)
- void **invalidate_link** (unsigned current_object, const P &start_point, const P &p)
- void **invalidate_link_** (unsigned current_object, const P &start_point, const P &p, anchor::Type anchor)
- **bool** **is_potential_link** (unsigned current_object, const P &start_point, const P &p) const
- **bool** **is_potential_link_** (unsigned current_object, const P &start_point, const P &p) const
- const L &**labeled_image** () const
- **unsigned** **link** (unsigned object) const
- const [object_links](#)< L > &**links** () const
- L::site **start_point** (unsigned current_object, anchor::Type anchor)
- L::site **start_point** (unsigned current_object)
- L::site **start_point_** (unsigned current_object, anchor::Type anchor)
- void **start_processing_object** (unsigned current_object)
- void **start_processing_object_** (unsigned current_object)
- **bool** **valid_link** (unsigned current_object, const P &start_point, const P &p, anchor::Type anchor)
- **bool** **valid_link_** (unsigned current_object, const P &start_point, const P &p, anchor::Type anchor)
- void **validate_link** (unsigned current_object, const P &start_point, const P &p, anchor::Type anchor)
- void **validate_link** (unsigned current_object, const P &start_point, const P &p)
- void **validate_link_** (unsigned current_object, const P &start_point, const P &p, anchor::Type anchor)
- **bool** **verify_link_criterion** (unsigned current_object, const P &start_point, const P &p, anchor::Type anchor) const
- **bool** **verify_link_criterion_** (unsigned current_object, const P &start_point, const P &p, anchor::Type anchor) const

Protected Attributes

- const [component_set](#)< L > **components_**
- anchor::Direction **direction_**
- **float** **dmax_**
- const F **dmax_f_**
- const L &**labeled_image_**
- [object_links](#)< L > **links_**

8.74.1 Detailed Description

template<typename L, typename F, typename E>class scribo::primitive::link::internal::link_single_dmax_ratio_base< L, F, E >

Base class for link functors using bounding box center and a proportional max distance.

Definition at line 72 of file link_single_dmax_ratio_base.hh.

8.74.2 Member Function Documentation

8.74.2.1 `template<typename L, typename E> void scribo::primitive::link::internal::link_func_base< L, E >::invalidate_link (unsigned current_object, const P & start_point, const P & p)` [inherited]

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.74.2.2 `template<typename L, typename E> L::site scribo::primitive::link::internal::link_func_base< L, E >::start_point (unsigned current_object)` [inherited]

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.74.2.3 `template<typename L, typename E> void scribo::primitive::link::internal::link_func_base< L, E >::validate_link (unsigned current_object, const P & start_point, const P & p)` [inherited]

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts. `anchor` is set to `anchor::MassCenter`.

8.75 scribo::Serializable< E > Class Template Reference

```
#include <scribo/core/concept/serializable.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef E **exact_t**

Public Member Functions

- `template<typename E2 >`
void `accept` (const `SerializeVisitor`< E2 > &visitor) const

8.75.1 Detailed Description

```
template<typename E>class scribo::Serializable< E >
```

Concept for objects that can be serialized.

Definition at line 42 of file `serializable.hh`.

8.75.2 Member Function Documentation

8.75.2.1 `template<typename E> template<typename E2 > void scribo::Serializable< E >::accept (const SerializeVisitor< E2 > & visitor)` const

Allow this object to be serialized by `visitor`.

8.76 scribo::SerializeVisitor< E > Class Template Reference

```
#include <scribo/core/concept/serialize_visitor.hh>
```

Public Types

- typedef **Object**< void > **category**
- typedef E **exact_t**

8.76.1 Detailed Description

```
template<typename E>class scribo::SerializeVisitor< E >
```

Link functor concept.

Definition at line 40 of file `serialize_visitor.hh`.

8.77 scribo::toolchain::internal::content_in_doc_functor< I > Struct Template Reference

```
#include <scribo/toolchain/internal/content_in_doc_functor.hh>
```

Public Types

- typedef **mln::trait::ch_value**
< I, V >::ret L
- typedef **scribo::def::lbl_type** V

Public Member Functions

- **content_in_doc_functor** (const char *doc_filename)
- virtual **int nsteps** () const
- virtual void **on_end** ()
- virtual void **on_new_progress_label** (const char *label)
- virtual void **on_progress** ()
- virtual void **on_start** ()
- virtual void **on_xml_saved** ()
- template<typename J >
[scribo::document](#)< L > **operator()** (const **Image**< J > &original_image, const **Image**< I > &processed_image)

Public Attributes

- [document](#)< L > **doc**
- **bool enable_denoising**
- **bool enable_line_seps**
- **bool enable_ocr**
- **bool enable_whitespace_seps**
- **mln::util::timer** gt
- std::string **ocr_language**
- std::string **output_file**
- **bool save_doc_as_xml**

- `mln::util::timer t`
- `bool verbose`
- `scribo::io::xml::Format xml_format`

8.77.1 Detailed Description

`template<typename I> struct scribo::toolchain::internal::content_in_doc_functor< I >`

Functor analysing and extracting document image content.

Definition at line 103 of file `content_in_doc_functor.hh`.

8.78 `scribo::toolchain::internal::content_in_hdoc_functor< I >` Struct Template Reference

```
#include <scribo/toolchain/internal/content_in_hdoc_functor.hh>
```

Public Types

- `typedef mln::trait::ch_value< I, V >::ret L`
- `typedef scribo::def::lbl_type V`

Public Member Functions

- `content_in_hdoc_functor (const char *doc_filename)`
- `virtual int nsteps () const`
- `virtual void on_end ()`
- `virtual void on_new_progress_label (const char *label)`
- `virtual void on_progress ()`
- `virtual void on_start ()`
- `virtual void on_xml_saved ()`
- `template<typename J> scribo::document< L > operator() (const Image< J > &original_image, const Image< I > &processed_image)`

Public Attributes

- `document< L > doc`
- `bool enable_denoising`
- `bool enable_line_seps`
- `bool enable_ocr`
- `bool enable_whitespace_seps`
- `mln::util::timer gt`
- `std::string ocr_language`
- `std::string output_file`
- `bool save_doc_as_xml`
- `mln::util::timer t`
- `bool verbose`
- `scribo::io::xml::Format xml_format`

8.78.1 Detailed Description

template<typename I>struct scribo::toolchain::internal::content_in_hdoc_functor< I >

Functor analysing and extracting content in degraded/historical documents.

Definition at line 116 of file content_in_hdoc_functor.hh.

8.79 scribo::toolchain::internal::text_in_doc_functor< I > Struct Template Reference

```
#include <scribo/toolchain/internal/text_in_doc_functor.hh>
```

Public Types

- typedef **mln::trait::ch_value**
 < I, V >::ret L
- typedef **scribo::def::lbl_type** V

Public Member Functions

- virtual **int nsteps** () const
- virtual void **on_end** ()
- virtual void **on_new_progress_label** (const char *label)
- virtual void **on_progress** ()
- virtual void **on_start** ()
- [line_set](#)< L > **operator()** (const **Image**< I > &input_)

Public Attributes

- **bool enable_denoising**
- **bool enable_line_seps**
- **bool enable_whitespace_seps**
- **mln::util::timer** gt
- **std::string** ocr_language
- [line_set](#)< L > **output**
- **mln::util::timer** t
- **bool verbose**

8.79.1 Detailed Description

template<typename I>struct scribo::toolchain::internal::text_in_doc_functor< I >

Functor extracting text lines from a document image.

Definition at line 87 of file text_in_doc_functor.hh.

8.80 scribo::toolchain::internal::text_in_doc_preprocess_functor< I > Struct Template Reference

```
#include <scribo/toolchain/internal/text_in_doc_preprocess_functor.hh>
```

Public Member Functions

- virtual **int** **nsteps** () const
- virtual void **on_end** ()
- virtual void **on_new_progress_label** (const char *label)
- virtual void **on_progress** ()
- virtual void **on_start** ()
- **mln::trait::ch_value**< I, bool >
::ret operator() (const **Image**< I > &input_)

Public Attributes

- **mln::trait::concrete**< I >::ret **bg**
- Binarization_Algo **binarization_algo**
- **bool** **enable_denoising**
- **bool** **enable_deskew**
- **bool** **enable_fg_extraction**
- **bool** **enable_subsample**
- **mln::trait::concrete**< I >::ret **fg**
- **mln::util::timer** **gt**
- **unsigned** **lambda**
- **image2d**< **bool** > **output**
- **double** **sauvola_k2**
- **double** **sauvola_k3**
- **double** **sauvola_k4**
- **unsigned** **sauvola_win**
- **mln::util::timer** **t**
- **bool** **verbose**

8.80.1 Detailed Description

`template<typename I>struct scribo::toolchain::internal::text_in_doc_preprocess_functor< I >`

Functor performing custom preprocessing algorithms on documents.

Whatever the options selected, this functor converts an input image into a graylevel image and try to binarize the latter.

Optional algorithms can be performed too:

- Subsampling (**enable_subsample**), disabled by default.
- Extract Background/Foreground (**enable_fg_extraction**), split background and foreground objects, disabled by default.
- Deskew (**enable_deskew**), disabled by default.
- Denoise (**enable_denoise**), enabled by default.

The binarization algorithm can be chosen through **binarization_algo** option. **It** can be one of the algorithms detailed in enum **Binarization_Algo**.

Few parameters can be set for algorithms:

- Sauvola Multi-scale: **sauvola_win**, the window size used in Sauvola based algorithms (default 101). **sauvola_K2**, a user parameter for Sauvola's threshold formula at scale 2. (default 0.34). **sauvola_K3**, a user parameter for Sauvola's threshold formula at scale 3. (default 0.34). **sauvola_K4**, a user parameter for Sauvola's threshold formula at scale 4. (default 0.34).

- Extract background/foreground lambda, the maximum area of the possible foreground objects (default 0, auto).

Definition at line 115 of file text_in_doc_preprocess_functor.hh.

8.81 scribo::toolchain::internal::text_in_picture_functor< I > Struct Template Reference

```
#include <scribo/toolchain/internal/text_in_picture_functor.hh>
```

Public Types

- typedef `mln::trait::ch_value`
 < I, V >::ret L
- typedef `scribo::def::lbl_type` V

Public Member Functions

- virtual `int nsteps` () const
- virtual void `on_end` ()
- virtual void `on_new_progress_label` (const char *label)
- virtual void `on_progress` ()
- virtual void `on_start` ()
- `component_set`< L > `operator`() (const `Image`< I > &)

Public Attributes

- `float bbox_h_ratio`
- `float bbox_overlap`
- `bool enable_bg_removal`
- `bool enable_multi_scale_bin`
- `unsigned group_min_holes`
- `object_groups`< L > `groups`
- `mln::util::timer` gt
- `unsigned lambda`
- `unsigned max_dim_size`
- `unsigned mean_width`
- `component_set`< L > `output`
- `unsigned regroup_dmax`
- `unsigned sauvola_min_w`
- `unsigned sauvola_s`
- `unsigned small_groups`
- `mln::util::timer` t
- `bool verbose`

8.81.1 Detailed Description

```
template<typename I>struct scribo::toolchain::internal::text_in_picture_functor< I >
```

Localize text in a picture.

Definition at line 112 of file text_in_picture_functor.hh.

8.82 scribo::toolchain::internal::Toolchain_Functor Class Reference

```
#include <scribo/toolchain/internal/toolchain_functor.hh>
```

Public Member Functions

- virtual **int** **nsteps** () const =0
- virtual void **on_end** ()
- virtual void **on_new_progress_label** (const char *label)
- virtual void **on_progress** ()
- virtual void **on_start** ()

Public Attributes

- **bool** **verbose**

8.82.1 Detailed Description

Base class for toolchain functors.

Definition at line 47 of file toolchain_functor.hh.

8.83 scribo::util::integral_sub_sum_sum2_functor< I, S > Class Template Reference

Public Types

- typedef **mln::util::couple**< S, S > **result**

Public Member Functions

- **integral_sub_sum_sum2_functor** (const I &ima, **unsigned** scale)
- **integral_sub_sum_sum2_functor** (**unsigned** scale, const mln_box(I)&output_domain, **unsigned** border)
- void **begin_of_col** ()
- void **begin_of_first_row** ()
- void **begin_of_row** ()
- void **end_of_col** ()
- void **end_of_row** ()
- void **take** (const V &v)
- **result to_result** (const **result** &up_result) const
- **result to_result_first_row** () const

Public Attributes

- **J** **sub**

8.83.1 Detailed Description

```
template<typename I, typename S = typename mln::value::props< typename I ::value >::sum> class scribo::util::integral_sub_sum_sum2_functor< I, S >
```

Definition at line 48 of file integral_sub_sum_sum2_functor.hh.

8.84 scribo::util::integral_sum_sum2_functor< V, S > Class Template Reference

Public Types

- typedef `mln::util::couple< S, S >` **result**

Public Member Functions

- void **begin_of_col** ()
- void **begin_of_first_row** ()
- void **begin_of_row** ()
- void **end_of_col** ()
- void **end_of_row** ()
- void **take** (const V &v)
- **result to_result** (const **result** &up_result) const
- **result to_result_first_row** () const

8.84.1 Detailed Description

```
template<typename V, typename S = typename mln::value::props< V >::sum>class scribo::util::integral_sum_sum2_functor<
V, S >
```

Definition at line 44 of file `integral_sum_sum2_functor.hh`.

8.85 scribo::util::integral_sum_sum2_global_min_functor< V, S > Class Template Reference

Public Types

- typedef `mln::util::couple< S, S >` **result**

Public Member Functions

- void **begin_of_col** ()
- void **begin_of_first_row** ()
- void **begin_of_row** ()
- void **end_of_col** ()
- void **end_of_row** ()
- const V & **global_min** () const
- void **take** (const V &v)
- **result to_result** (const **result** &up_result) const
- **result to_result_first_row** () const

8.85.1 Detailed Description

```
template<typename V, typename S = typename mln::value::props< V >::sum>class scribo::util::integral_sum_sum2_global_min-
_functor< V, S >
```

Definition at line 44 of file `integral_sum_sum2_global_min_functor.hh`.

8.86 stats< T > Class Template Reference

Public Member Functions

- **stats** (const int size)
- util::array< [cluster_stats](#)< T > > & **clusters** ()
- **T max** ()
- **T mean** ()
- **T median** ()
- **T min** ()
- **unsigned nelements** ()
- void **reset** ()
- **T standard_deviation** ()
- void **take** (const T &value)
- **T variance** ()

8.86.1 Detailed Description

template<typename T>class stats< T >

Definition at line 195 of file stats.hh.

Bibliography

- [1] E. Badekas and N. Papamarkos. Automatic evaluation of document binarization results. pages 1005–1014, 2005.
- [2] Boontee Kruatrachue, Narongchai Moongfangklang, and Kritawan Siriboon. Fast document segmentation using contour and XY-Cut technique. In *Proceedings of World Academy of Science, Engineering and Technology*, volume 5, pages 27–29, 2005.
- [3] J.-L. Meunier. Optimized XY-Cut for determining a page reading order. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 347–351. IEEE, 2005.
- [4] George Nagy, Sharad Seth, and Mahesh Viswanathan. A prototype document image analysis system for technical journals. *Computer*, 25(7):10–22, 1992.
- [5] J. Sauvola and M. Pietikäinen. Adaptive document image binarization. *PATTERN RECOGNITION*, 33:225–236, 2000.
- [6] Faisal Shafait, Daniel Keysers, and Thomas M. Breuel. Efficient implementation of local adaptive thresholding techniques using integral images. *Document Recognition and Retrieval XV*, Jan 2008.

Index

accept

- scribo::component_info, 150
- scribo::component_set, 153
- scribo::document, 160
- scribo::line_info, 178
- scribo::line_links, 180
- scribo::object_groups, 184
- scribo::object_links, 185
- scribo::paragraph_set, 188
- scribo::Serializable, 202

add_line

- scribo::paragraph_info, 188

add_separators

- scribo::component_set, 153

align_lines

- scribo::table::internal, 135

align_lines_horizontally

- scribo::table, 132

align_lines_vertically

- scribo::table, 132

alignment_decision_image

- Debug, 33

alignments

- scribo::primitive::extract, 114

apply

- scribo::primitive::group, 116

b_

- scribo::fun::v2b::components_on_border_filter, 163

bboxes_enlarged_image

- Debug, 33

bboxes_image

- Debug, 34

Binarization, 17

bounding_box_links

- scribo::draw, 102

bounding_boxes

- scribo::draw, 103

canvas

- scribo::primitive::extract, 114

cells

- scribo::primitive::extract, 115

central_sites

- scribo, 91

char_space_image

- Debug, 34

Character

- scribo::component, 97

clean

- scribo::text, 137

clean_inplace

- scribo::text, 137

clear

- scribo::object_links, 186

clear_separators

- scribo::component_set, 153

cluster_stats< T >, 139

comp_to_link

- scribo::object_links, 186

compare_values< T >, 139

Component Filtering, 48

- components_large, 49
- components_on_border, 49
- components_small, 50
- components_thin, 50
- components_v_thin, 51
- components_with_two_holes, 51
- objects_h_thick, 51, 52
- objects_h_thin, 52
- objects_size_ratio, 52
- objects_thick, 53
- objects_v_thick, 53, 54
- objects_v_thin, 54
- objects_with_holes, 54

Component Group Filtering, 40

- object_groups_mean_width, 40
- object_groups_size_ratio, 40
- object_groups_small, 40
- object_groups_with_holes, 41

Component Link Filtering, 42

- object_links_aligned, 42
- object_links_bbox_h_ratio, 43
- object_links_bbox_overlap, 43
- object_links_bbox_ratio, 43
- object_links_bbox_w_ratio, 44
- object_links_bottom_aligned, 44
- object_links_center_aligned, 44
- object_links_left_aligned, 45
- object_links_non_aligned_simple, 45
- object_links_right_aligned, 46
- object_links_top_aligned, 46

component_id_t

- scribo, 91

component_set

- scribo::component_set, 152, 153

components

- Components Extraction, 63

- scribo::line_set, 181
- scribo::object_links, 186
- Components Extraction, 63
 - components, 63
- components_
 - scribo::fun::v2b::components_large_filter, 162
 - scribo::fun::v2b::components_on_border_filter, 163
 - scribo::fun::v2b::components_small_filter, 165
 - scribo::line_set, 181
- components_large
 - Component Filtering, 49
- components_large_filter
 - scribo::fun::v2b::components_large_filter, 161
- components_on_border
 - Component Filtering, 49
- components_on_border_filter
 - scribo::fun::v2b::components_on_border_filter, 163
- components_small
 - Component Filtering, 50
- components_small_filter
 - scribo::fun::v2b::components_small_filter, 164
- components_thin
 - Component Filtering, 50
- components_v_thin
 - Component Filtering, 51
- components_with_two_holes
 - Component Filtering, 51
- compute
 - scribo::primitive::link, 122
- compute_lines
 - scribo::line_set, 181
- compute_several
 - scribo::primitive::link, 123
- connect_horizontal_lines
 - scribo::table, 132
- connect_lines
 - scribo::table::internal, 135
- connect_vertical_lines
 - scribo::table, 133
- content_in_doc
 - Processing, 87
- content_in_hdoc
 - Processing, 88
- crop
 - scribo::preprocessing, 108
- crop_without_localization
 - scribo::preprocessing, 109
- Data structures, 30
- Debug, 31
 - alignment_decision_image, 33
 - bboxes_enlarged_image, 33
 - bboxes_image, 34
 - char_space_image, 34
 - decision_image, 34
 - highlight_text_area, 35
 - line_info_image, 36
 - linked_bboxes_image, 36, 37
 - links_decision_image, 37
 - links_image, 37
 - logger, 37
 - looks_like_a_text_line_image, 37, 38
 - mean_and_base_lines_image, 38
 - save_comp_diff, 38
 - save_label_image, 39
 - save_table_image, 39
 - text_areas_image, 39
 - text_color_image, 39
- debug_filename
 - scribo::make, 106
- decision_image
 - Debug, 34
- default_verbose_mode
 - scribo::debug::internal::logger_, 155
- delta_of_line
 - scribo::line_info, 178
- denoise
 - scribo::preprocessing, 109
- denoise_bg
 - scribo::preprocessing, 109
- denoise_fg
 - scribo::preprocessing, 109
- deskw
 - scribo::preprocessing, 110
- disable_linking
 - scribo::object_links, 186
- Documents, 82
- DropCapital
 - scribo::component, 97
- duplicate
 - scribo::component_set, 153
 - scribo::line_set, 181
 - scribo::object_links, 186
- ebbox
 - scribo::line_info, 178
- Element Filtering, 55, 58
 - images_in_paragraph, 55
 - objects_in_borders, 55
 - separators_in_borders, 56
 - separators_in_element, 56
 - separators_in_paragraph, 56
 - separators_vert_in_borders, 56
- erase
 - scribo::table, 133
- erase_objects
 - scribo, 92
- extract
 - scribo::table, 133
- extract_lines
 - Text Extraction, 77
- extract_lines_with_features
 - Text Extraction, 77, 78
- extract_lines_wo_merge
 - Text Extraction, 78
- extract_paragraphs

- Text Extraction, 78
- extract_paragraphs_hdoc
 - Text Extraction, 78
- fast_merge
 - scribo::line_info, 178
- filename_prefix
 - scribo::debug::internal::logger_, 155
- fill_object_holes
 - scribo::postprocessing, 107
- find_graph_link
 - scribo::primitive::internal, 118
- find_link
 - scribo::primitive::internal, 118
- find_root
 - scribo::primitive::internal, 119
- force_stats_update
 - scribo::line_info, 178
 - scribo::line_set, 181
- from_double_link
 - scribo::primitive::group, 116
- from_double_link_any
 - scribo::primitive::group, 117
- from_graph
 - scribo::primitive::group, 117
- from_single_link
 - scribo::primitive::group, 117
- global_threshold
 - scribo::binarization, 94
- global_threshold_auto
 - scribo::binarization, 94
- group_of
 - scribo::object_groups, 184
- groups
 - scribo::line_set, 181
- groups_bboxes
 - scribo::draw, 103
- has_linking_enabled
 - scribo::object_links, 186
- has_separators
 - scribo::component_set, 153
- has_text
 - scribo::document, 160
- have_link_valid
 - scribo::primitive::internal, 119
- highlight_text_area
 - Debug, 35
- homogeneous_contrast
 - scribo::preprocessing, 110
- HorizontalLineSeparator
 - scribo::component, 97
- horizontal_separators
 - Lines and Separators extraction, 67
- id
 - scribo::line_info, 179
- Ignored
 - scribo::component, 97
- Image
 - scribo::component, 97
- images_in_paragraph
 - Element Filtering, 55
- images_to_drop_capital
 - scribo::postprocessing, 107
- influence_zone_graph
 - scribo::make, 106
- info
 - scribo::component_set, 153
 - scribo::line_set, 182
- infos
 - scribo::line_set, 182
- infos_
 - scribo::line_set, 182
- init
 - scribo::object_links, 186
- init_
 - scribo::component_set, 153
- invalidate_link
 - scribo::primitive::link::internal::link_functor_base, 193
 - scribo::primitive::link::internal::link_several_dmax_base, 195
 - scribo::primitive::link::internal::link_single_dmax_base, 196
 - scribo::primitive::link::internal::link_single_dmax_ratio_aligned_base, 198
 - scribo::primitive::link::internal::link_single_dmax_ratio_aligned_delta_base, 200
 - scribo::primitive::link::internal::link_single_dmax_ratio_base, 202
- is_at_level
 - scribo::debug::internal::logger_, 156
- is_at_verbose_mode
 - scribo::debug::internal::logger_, 156
- is_enabled
 - scribo::debug::internal::logger_, 156
- is_hidden
 - scribo::line_info, 179
- is_link_valid
 - scribo::primitive::internal, 119
- is_linked
 - scribo::object_links, 186
- is_valid
 - scribo::component_set, 154
 - scribo::line_set, 182
 - scribo::object_links, 186
- is_verbose
 - scribo::debug::internal::logger_, 156
- Kim, 18
 - kim, 18
- kim
 - Kim, 18
- labeled_image
 - scribo::component_set, 154

- Layout Analysis, 62
 - xy_cut, 62
- Level
 - scribo::debug, 101
- level
 - scribo::debug::internal::logger_, 156
- Line Link Filtering, 59
 - line_links_x_height, 59
- line_info
 - scribo::line_info, 178
- line_info_image
 - Debug, 36
- line_links_x_height
 - Line Link Filtering, 59
- line_set
 - scribo::line_set, 181
- Lines and Separators extraction, 66
 - horizontal_separators, 67
 - lines_discontinued, 67
 - lines_h_discontinued, 68
 - lines_h_pattern, 68
 - lines_h_single, 69
 - lines_h_thick_and_single, 69
 - lines_h_thick_and_thin, 70
 - lines_pattern, 70
 - lines_thick, 70
 - lines_v_discontinued, 71
 - lines_v_pattern, 71
 - lines_v_single, 71, 72
 - lines_v_thick_and_single, 72
 - separators, 72
 - separators_nonvisible, 73
 - vertical_separators, 73
- lines_discontinued
 - Lines and Separators extraction, 67
- lines_h_discontinued
 - Lines and Separators extraction, 68
- lines_h_pattern
 - Lines and Separators extraction, 68
- lines_h_single
 - Lines and Separators extraction, 69
- lines_h_thick_and_single
 - Lines and Separators extraction, 69
- lines_h_thick_and_thin
 - Lines and Separators extraction, 70
- lines_pattern
 - Lines and Separators extraction, 70
- lines_thick
 - Lines and Separators extraction, 70
- lines_v_discontinued
 - Lines and Separators extraction, 71
- lines_v_pattern
 - Lines and Separators extraction, 71
- lines_v_single
 - Lines and Separators extraction, 71, 72
- lines_v_thick_and_single
 - Lines and Separators extraction, 72
- link_funcutor_base
 - scribo::primitive::link::internal::link_funcutor_base, 193
- link_lines
 - Text Extraction, 79
- linked_bboxes_image
 - Debug, 36, 37
- links
 - scribo::line_set, 182
- links_decision_image
 - Debug, 37
- links_image
 - Debug, 37
- local_threshold
 - scribo::binarization, 95
- log
 - scribo::debug::internal::logger_, 156
- log_image
 - scribo::debug::internal::logger_, 156
- logger
 - Debug, 37
- look_like_text_lines
 - scribo::text, 137
- look_like_text_lines_inplace
 - scribo::text, 137
- looks_like_a_text_line_image
 - Debug, 37, 38
- marked_
 - scribo::fun::v2b::components_large_filter, 162
 - scribo::fun::v2b::components_small_filter, 165
- max_size_
 - scribo::fun::v2b::components_large_filter, 162
- mean_and_base_lines_image
 - Debug, 38
- merge_double_link
 - scribo::primitive::link, 123
- merge_double_link_closest_aligned
 - scribo::primitive::link, 123
- merging
 - Text Extraction, 79
- merging_hdoc
 - Text Extraction, 79
- min_size_
 - scribo::fun::v2b::components_small_filter, 165
- mIn::info, 140
- nelements
 - scribo::component_set, 154
 - scribo::line_set, 182
 - scribo::object_links, 186
- Niblack, 19
 - niblack, 19, 20
 - niblack_fast, 20
 - niblack_threshold, 20, 21
- niblack
 - Niblack, 19, 20
- niblack_fast
 - Niblack, 20
- niblack_threshold

- Niblack, 20, 21
- nlabels_
 - scribo::fun::v2b::components_large_filter, 162
 - scribo::fun::v2b::components_small_filter, 165
- Noise
 - scribo::component, 97
- non_text
 - scribo::primitive::extract, 115
- non_text_hdoc
 - scribo::primitive::extract, 115
- None
 - scribo::component, 97
- object_groups_mean_width
 - Component Group Filtering, 40
- object_groups_size_ratio
 - Component Group Filtering, 40
- object_groups_small
 - Component Group Filtering, 40
- object_groups_with_holes
 - Component Group Filtering, 41
- object_links
 - scribo::object_links, 185
- object_links_aligned
 - Component Link Filtering, 42
- object_links_bbox_h_ratio
 - Component Link Filtering, 43
- object_links_bbox_overlap
 - Component Link Filtering, 43
- object_links_bbox_ratio
 - Component Link Filtering, 43
- object_links_bbox_w_ratio
 - Component Link Filtering, 44
- object_links_bottom_aligned
 - Component Link Filtering, 44
- object_links_center_aligned
 - Component Link Filtering, 44
- object_links_left_aligned
 - Component Link Filtering, 45
- object_links_non_aligned_simple
 - Component Link Filtering, 45
- object_links_right_aligned
 - Component Link Filtering, 46
- object_links_top_aligned
 - Component Link Filtering, 46
- objects_h_thick
 - Component Filtering, 51, 52
- objects_h_thin
 - Component Filtering, 52
- objects_in_borders
 - Element Filtering, 55
- objects_size_ratio
 - Component Filtering, 52
- objects_thick
 - Component Filtering, 53
- objects_v_thick
 - Component Filtering, 53, 54
- objects_v_thin
 - Component Filtering, 54
- objects_with_holes
 - Component Filtering, 54
- operator<<
 - scribo::component, 97
 - scribo::component_set, 154
 - scribo::debug::internal::logger_, 156
 - scribo::object_links, 187
- operator()
 - scribo::binarization::internal::niblack_formula, 140
 - scribo::binarization::internal::sauvola_formula, 143
 - scribo::binarization::internal::singh_formula, 146
 - scribo::binarization::internal::wolf_formula, 147
 - scribo::component_set, 154
 - scribo::fun::v2b::components_large_filter, 161
 - scribo::fun::v2b::components_on_border_filter, 163
 - scribo::fun::v2b::components_small_filter, 164
 - scribo::line_set, 182
 - scribo::object_groups, 184
 - scribo::object_links, 186
- operator=
 - scribo::line_info, 179
- operator==
 - scribo::component_set, 154
- Otsu, 22
 - otsu, 22
 - otsu_threshold, 22
- otsu
 - Otsu, 22
- otsu_threshold
 - Otsu, 22
- paragraph
 - scribo::make, 106
- Paragraph Filtering, 60
 - paragraphs_bbox_overlap, 60
 - paragraphs_in_borders, 60
 - paragraphs_in_image, 60
- paragraphs_bbox_overlap
 - Paragraph Filtering, 60
- paragraphs_in_borders
 - Paragraph Filtering, 60
- paragraphs_in_image
 - Paragraph Filtering, 60
- Pictures, 81
- precise_merge
 - scribo::line_info, 179
- Preprocessing, 83, 84
 - text_in_doc_preprocess, 84, 85
- Primitive Extraction, 65
- Processing, 86, 87
 - content_in_doc, 87
 - content_in_hdoc, 88
 - text_in_doc, 88
 - text_in_picture, 86
- Punctuation
 - scribo::component, 97
- rd

- scribo::primitive::internal, 119
- rebuild
 - scribo::table, 133
- recognition
 - Text Recognition, 75
- repair_horizontal_lines
 - scribo::table, 134
- repair_lines
 - scribo::table::internal, 135
- repair_vertical_lines
 - scribo::table, 134
- rotate_90
 - scribo::preprocessing, 110, 111
- Routines, 74
- Sauvola, 23
 - sauvola, 24, 25
 - sauvola_ms, 25, 26
 - sauvola_ms_split, 26
 - sauvola_threshold, 26, 27
- sauvola
 - Sauvola, 24, 25
- sauvola_ms
 - Sauvola, 25, 26
 - scribo::binarization, 95
- sauvola_ms_split
 - Sauvola, 26
- sauvola_threshold
 - Sauvola, 26, 27
- save_comp_diff
 - Debug, 38
- save_label_image
 - Debug, 39
- save_table_image
 - Debug, 39
- scribo, 89
 - central_sites, 91
 - component_id_t, 91
 - erase_objects, 92
- scribo::component
 - Character, 97
 - DropCapital, 97
 - HorizontalLineSeparator, 97
 - Ignored, 97
 - Image, 97
 - Noise, 97
 - None, 97
 - Punctuation, 97
 - Undefined, 97
 - VerticalLineSeparator, 97
 - WhitespaceSeparator, 97
- scribo::DMax_Functor< E >, 158
- scribo::Link_Functor< E >, 183
- scribo::Serializable
 - accept, 202
- scribo::Serializable< E >, 202
- scribo::SerializeVisitor< E >, 203
- scribo::binarization, 92
 - global_threshold, 94
 - global_threshold_auto, 94
 - local_threshold, 95
 - sauvola_ms, 95
 - singh, 95, 96
- scribo::binarization::internal::niblack_formula, 140
 - operator(), 140
- scribo::binarization::internal::niblack_functor< I >, 141
- scribo::binarization::internal::niblack_functor_fast< I >, 141
- scribo::binarization::internal::niblack_threshold_functor< I >, 142
- scribo::binarization::internal::sauvola_formula, 143
 - operator(), 143
- scribo::binarization::internal::sauvola_functor< I >, 143
- scribo::binarization::internal::sauvola_ms_functor< I >, 144
- scribo::binarization::internal::sauvola_threshold_functor< I >, 145
- scribo::binarization::internal::singh_formula
 - operator(), 146
- scribo::binarization::internal::singh_formula< V >, 145
- scribo::binarization::internal::singh_functor< I >, 146
- scribo::binarization::internal::wolf_formula
 - operator(), 147
- scribo::binarization::internal::wolf_formula< V >, 147
- scribo::binarization::internal::wolf_functor< I >, 147
- scribo::binarization::internal::wolf_functor_fast< I >, 148
- scribo::component, 96
 - operator<<, 97
 - str2tag, 97
 - str2type, 98
 - Tag, 97
 - Type, 97
- scribo::component_features_data, 149
- scribo::component_info
 - accept, 150
- scribo::component_info< L >, 149
- scribo::component_set
 - accept, 153
 - add_separators, 153
 - clear_separators, 153
 - component_set, 152, 153
 - duplicate, 153
 - has_separators, 153
 - info, 153
 - init_, 153
 - is_valid, 154
 - labeled_image, 154
 - nelements, 154
 - operator<<, 154
 - operator(), 154
 - operator==, 154
 - separators, 154
 - update_tags, 154
- scribo::component_set< L >, 150
- scribo::core, 98
- scribo::debug, 98

- Level, 101
- txt_to_verbose_mode, 101
- usage, 101
- VerboseMode, 101
- scribo::debug::arg_data, 154
- scribo::debug::internal::logger_, 155
 - default_verbose_mode, 155
 - filename_prefix, 155
 - is_at_level, 156
 - is_at_verbose_mode, 156
 - is_enabled, 156
 - is_verbose, 156
 - level, 156
 - log, 156
 - log_image, 156
 - operator<<, 156
 - set_default_verbose_mode, 156
 - set_filename_prefix, 156
 - set_level, 156
 - set_verbose_mode, 156
 - set_verbose_prefix, 157
 - start_time_logging, 157
 - stop_time_logging, 157
 - verbose_mode, 157
- scribo::debug::opt_data, 157
- scribo::debug::option_parser, 157
- scribo::debug::toggle_data, 158
- scribo::doc_serializer< E >, 158
- scribo::document
 - accept, 160
 - has_text, 160
- scribo::document< L >, 159
- scribo::draw, 101
 - bounding_box_links, 102
 - bounding_boxes, 103
 - groups_bboxes, 103
- scribo::filter, 103
- scribo::fun::v2b::components_large_filter
 - components_, 162
 - components_large_filter, 161
 - marked_, 162
 - max_size_, 162
 - nlabels_, 162
 - operator(), 161
- scribo::fun::v2b::components_large_filter< L >, 160
- scribo::fun::v2b::components_on_border_filter
 - b_, 163
 - components_, 163
 - components_on_border_filter, 163
 - operator(), 163
- scribo::fun::v2b::components_on_border_filter< L >, 162
- scribo::fun::v2b::components_small_filter
 - components_, 165
 - components_small_filter, 164
 - marked_, 165
 - min_size_, 165
 - nlabels_, 165
 - operator(), 164
- scribo::fun::v2b::components_small_filter< L >, 164
- scribo::fun::v2b::label_to_bool< L >, 165
- scribo::fun::v2v::highlight< R >, 166
- scribo::group_info, 166
- scribo::internal::component_set_data< L >, 167
- scribo::internal::document_data< L >, 167
- scribo::internal::line_info_data< L >, 168
- scribo::internal::line_links_data< L >, 169
- scribo::internal::line_set_data< L >, 169
- scribo::internal::object_groups_data< L >, 170
- scribo::internal::object_links_data< L >, 170
- scribo::internal::paragraph_set_data< L >, 171
- scribo::internal::sort_comp_ids< L >, 171
- scribo::io::img::internal::debug_img_visitor< L >, 172
- scribo::io::img::internal::full_img_visitor< L >, 172
- scribo::io::img::internal::non_text_img_visitor, 173
- scribo::io::img::internal::text_img_visitor, 173
- scribo::io::xml::internal::color_t, 174
- scribo::io::xml::internal::extended_page_xml_visitor< L >, 174
- scribo::io::xml::internal::full_xml_visitor, 175
- scribo::io::xml::internal::page_xml_visitor< L >, 175
- scribo::layout::internal::hist_info, 176
- scribo::layout::internal::node< B >, 176
- scribo::line_info
 - accept, 178
 - delta_of_line, 178
 - ebbox, 178
 - fast_merge, 178
 - force_stats_update, 178
 - id, 179
 - is_hidden, 179
 - line_info, 178
 - operator=, 179
 - precise_merge, 179
 - set_hidden, 179
 - update_ebbox, 179
- scribo::line_info< L >, 176
- scribo::line_links
 - accept, 180
- scribo::line_links< L >, 179
- scribo::line_set
 - components, 181
 - components_, 181
 - compute_lines, 181
 - duplicate, 181
 - force_stats_update, 181
 - groups, 181
 - info, 182
 - infos, 182
 - infos_, 182
 - is_valid, 182
 - line_set, 181
 - links, 182
 - nelements, 182
 - operator(), 182
 - update_tags, 182

- update_types, 182
- scribo::line_set< L >, 180
- scribo::make, 105
 - debug_filename, 106
 - influence_zone_graph, 106
 - paragraph, 106
 - text_blocks_image, 106
 - text_components_image, 107
- scribo::object_groups
 - accept, 184
 - group_of, 184
 - operator(), 184
- scribo::object_groups< L >, 183
- scribo::object_links
 - accept, 185
 - clear, 186
 - comp_to_link, 186
 - components, 186
 - disable_linking, 186
 - duplicate, 186
 - has_linking_enabled, 186
 - init, 186
 - is_linked, 186
 - is_valid, 186
 - nelements, 186
 - object_links, 185
 - operator<<, 187
 - operator(), 186
 - update, 187
- scribo::object_links< L >, 184
- scribo::paragraph_info
 - add_line, 188
- scribo::paragraph_info< L >, 187
- scribo::paragraph_set
 - accept, 188
- scribo::paragraph_set< L >, 188
- scribo::postprocessing, 107
 - fill_object_holes, 107
 - images_to_drop_capital, 107
- scribo::preprocessing, 107
 - crop, 108
 - crop_without_localization, 109
 - denoise, 109
 - denoise_bg, 109
 - denoise_fg, 109
 - deskew, 110
 - homogeneous_contrast, 110
 - rotate_90, 110, 111
 - split_bg_fg, 111
- scribo::preprocessing::internal::Hough, 189
- scribo::preprocessing::internal::QCompare, 189
- scribo::preprocessing::internal::s_angle, 189
- scribo::primitive, 111
- scribo::primitive::extract, 112
 - alignments, 114
 - canvas, 114
 - cells, 115
 - non_text, 115
 - non_text_hdoc, 115
- scribo::primitive::group, 116
 - apply, 116
 - from_double_link, 116
 - from_double_link_any, 117
 - from_graph, 117
 - from_single_link, 117
- scribo::primitive::internal, 118
 - find_graph_link, 118
 - find_link, 118
 - find_root, 119
 - have_link_valid, 119
 - is_link_valid, 119
 - rd, 119
 - update_graph_link, 119
- scribo::primitive::link, 120
 - compute, 122
 - compute_several, 123
 - merge_double_link, 123
 - merge_double_link_closest_aligned, 123
 - with_graph, 123
 - with_rag, 124
 - with_several_graphes, 124
 - with_several_left_links, 124
 - with_several_right_closest_links, 124
 - with_several_right_links, 125
 - with_several_right_links_overlap, 125
 - with_single_down_link, 125, 126
 - with_single_left_link, 126
 - with_single_left_link_dmax_ratio, 126, 127
 - with_single_left_link_dmax_ratio_aligned, 127
 - with_single_right_link, 128
 - with_single_right_link_bottom, 128
 - with_single_right_link_dmax_ratio, 128, 129
 - with_single_right_link_dmax_ratio_aligned, 129, 130
 - with_single_right_link_top, 130
 - with_single_up_link, 130, 131
- scribo::primitive::link::internal::dmax_default, 189
- scribo::primitive::link::internal::dmax_functor_base< E >, 190
- scribo::primitive::link::internal::dmax_hrules, 190
- scribo::primitive::link::internal::dmax_width_and_height, 191
- scribo::primitive::link::internal::dmax_width_only, 191
- scribo::primitive::link::internal::link_functor_base
 - invalidate_link, 193
 - link_functor_base, 193
 - start_point, 193
 - validate_link, 193
- scribo::primitive::link::internal::link_functor_base< L, E >, 192
- scribo::primitive::link::internal::link_several_dmax_base
 - invalidate_link, 195
 - start_point, 195
 - validate_link, 195
- scribo::primitive::link::internal::link_several_dmax_base< L, E >, 194

- scribo::primitive::link::internal::link_single_dmax_base
 - invalidate_link, 196
 - start_point, 196
 - validate_link, 197
- scribo::primitive::link::internal::link_single_dmax_base< L, E >, 195
- scribo::primitive::link::internal::link_single_dmax_ratio_
 - aligned_base
 - invalidate_link, 198
 - start_point, 198
 - validate_link, 198
- scribo::primitive::link::internal::link_single_dmax_ratio_
 - aligned_base< L, F, E >, 197
- scribo::primitive::link::internal::link_single_dmax_ratio_
 - aligned_delta_base
 - invalidate_link, 200
 - start_point, 200
 - validate_link, 200
- scribo::primitive::link::internal::link_single_dmax_ratio_
 - aligned_delta_base< L, F, E >, 199
- scribo::primitive::link::internal::link_single_dmax_ratio_
 - base
 - invalidate_link, 202
 - start_point, 202
 - validate_link, 202
- scribo::primitive::link::internal::link_single_dmax_ratio_
 - base< L, F, E >, 200
- scribo::table, 131
 - align_lines_horizontally, 132
 - align_lines_vertically, 132
 - connect_horizontal_lines, 132
 - connect_vertical_lines, 133
 - erase, 133
 - extract, 133
 - rebuild, 133
 - repair_horizontal_lines, 134
 - repair_vertical_lines, 134
- scribo::table::internal, 134
 - align_lines, 135
 - connect_lines, 135
 - repair_lines, 135
- scribo::text, 135
 - clean, 137
 - clean_inplace, 137
 - look_like_text_lines, 137
 - look_like_text_lines_inplace, 137
- scribo::toolchain::internal::Toolchain_Functor, 208
- scribo::toolchain::internal::content_in_doc_functor< I >, 203
- scribo::toolchain::internal::content_in_hdoc_functor< I >, 204
- scribo::toolchain::internal::text_in_doc_functor< I >, 205
- scribo::toolchain::internal::text_in_doc_preprocess_
 - functor< I >, 205
- scribo::toolchain::internal::text_in_picture_functor< I >, 207
- scribo::util::integral_sub_sum_sum2_functor< I, S >, 208
- scribo::util::integral_sum_sum2_functor< V, S >, 209
- scribo::util::integral_sum_sum2_global_min_functor< V, S >, 209
- separators
 - Lines and Separators extraction, 72
 - scribo::component_set, 154
- separators_in_borders
 - Element Filtering, 56
- separators_in_element
 - Element Filtering, 56
- separators_in_paragraph
 - Element Filtering, 56
- separators_nonvisible
 - Lines and Separators extraction, 73
- separators_vert_in_borders
 - Element Filtering, 56
- set_default_verbose_mode
 - scribo::debug::internal::logger_, 156
- set_filename_prefix
 - scribo::debug::internal::logger_, 156
- set_hidden
 - scribo::line_info, 179
- set_level
 - scribo::debug::internal::logger_, 156
- set_verbose_mode
 - scribo::debug::internal::logger_, 156
- set_verbose_prefix
 - scribo::debug::internal::logger_, 157
- singh
 - scribo::binarization, 95, 96
- split_bg_fg
 - scribo::preprocessing, 111
- start_point
 - scribo::primitive::link::internal::link_functor_base, 193
 - scribo::primitive::link::internal::link_several_dmax_
 - base, 195
 - scribo::primitive::link::internal::link_single_dmax_
 - base, 196
 - scribo::primitive::link::internal::link_single_dmax_
 - ratio_aligned_base, 198
 - scribo::primitive::link::internal::link_single_dmax_
 - ratio_aligned_delta_base, 200
 - scribo::primitive::link::internal::link_single_dmax_
 - ratio_base, 202
- start_time_logging
 - scribo::debug::internal::logger_, 157
- stats< T >, 210
- stop_time_logging
 - scribo::debug::internal::logger_, 157
- str2tag
 - scribo::component, 97
- str2type
 - scribo::component, 98
- Tag
 - scribo::component, 97

- Text Extraction, 76
 - extract_lines, 77
 - extract_lines_with_features, 77, 78
 - extract_lines_wo_merge, 78
 - extract_paragraphs, 78
 - extract_paragraphs_hdoc, 78
 - link_lines, 79
 - merging, 79
 - merging_hdoc, 79
- Text Recognition, 75
 - recognition, 75
- text_areas_image
 - Debug, 39
- text_blocks_image
 - scribo::make, 106
- text_color_image
 - Debug, 39
- text_components_image
 - scribo::make, 107
- text_extraction
 - Toolchains, 80
- text_in_doc
 - Processing, 88
- text_in_doc_preprocess
 - Preprocessing, 84, 85
- text_in_picture
 - Processing, 86
- Toolchains, 80
 - text_extraction, 80
- txt_to_verbose_mode
 - scribo::debug, 101
- Type
 - scribo::component, 97
- Undefined
 - scribo::component, 97
- update
 - scribo::object_links, 187
- update_ebbox
 - scribo::line_info, 179
- update_graph_link
 - scribo::primitive::internal, 119
- update_tags
 - scribo::component_set, 154
 - scribo::line_set, 182
- update_types
 - scribo::line_set, 182
- usage
 - scribo::debug, 101
- validate_link
 - scribo::primitive::link::internal::link_functor_base, 193
 - scribo::primitive::link::internal::link_several_dmax_base, 195
 - scribo::primitive::link::internal::link_single_dmax_base, 197
 - scribo::primitive::link::internal::link_single_dmax_ratio_aligned_base, 198
 - scribo::primitive::link::internal::link_single_dmax_ratio_aligned_delta_base, 200
 - scribo::primitive::link::internal::link_single_dmax_ratio_base, 202
- verbose_mode
 - scribo::debug::internal::logger_, 157
- VerboseMode
 - scribo::debug, 101
- VerticalLineSeparator
 - scribo::component, 97
- vertical_separators
 - Lines and Separators extraction, 73
- WhitespaceSeparator
 - scribo::component, 97
- with_graph
 - scribo::primitive::link, 123
- with_rag
 - scribo::primitive::link, 124
- with_several_graphes
 - scribo::primitive::link, 124
- with_several_left_links
 - scribo::primitive::link, 124
- with_several_right_closest_links
 - scribo::primitive::link, 124
- with_several_right_links
 - scribo::primitive::link, 125
- with_several_right_links_overlap
 - scribo::primitive::link, 125
- with_single_down_link
 - scribo::primitive::link, 125, 126
- with_single_left_link
 - scribo::primitive::link, 126
- with_single_left_link_dmax_ratio
 - scribo::primitive::link, 126, 127
- with_single_left_link_dmax_ratio_aligned
 - scribo::primitive::link, 127
- with_single_right_link
 - scribo::primitive::link, 128
- with_single_right_link_bottom
 - scribo::primitive::link, 128
- with_single_right_link_dmax_ratio
 - scribo::primitive::link, 128, 129
- with_single_right_link_dmax_ratio_aligned
 - scribo::primitive::link, 129, 130
- with_single_right_link_top
 - scribo::primitive::link, 130
- with_single_up_link
 - scribo::primitive::link, 130, 131
- Wolf, 28
 - wolf, 28, 29
 - wolf_fast, 29
- wolf
 - Wolf, 28, 29
- wolf_fast
 - Wolf, 29
- xy_cut
 - Layout Analysis, 62