

Markov Model of Keyboard Tokens

0

Generated by Doxygen 1.8.11

Contents

1	Namespace Index	1
1.1	Packages	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Namespace Documentation	9
5.1	Package components	9
5.2	Package data_analysis	9
5.3	Package gui	10
5.4	Package junit	10
5.5	Package rank	10
5.6	Package runtime	10
5.7	Package test	11
5.8	Package trie	11

6	Class Documentation	13
6.1	components.Chain Class Reference	13
6.1.1	Detailed Description	14
6.1.2	Constructor & Destructor Documentation	14
6.1.2.1	Chain(int window, int token, int threshold, int model_size)	14
6.1.2.2	Chain(Chain c)	14
6.1.3	Member Function Documentation	14
6.1.3.1	add_touch(Touch touch)	14
6.1.3.2	add_touch_list(List< Touch > t)	15
6.1.3.3	compare_to(Chain auth_chain)	15
6.1.3.4	compute_uncomputed()	16
6.1.3.5	get_distribution()	17
6.1.3.6	get_key_distribution()	17
6.1.3.7	get_model_size()	17
6.1.3.8	get_threshold()	18
6.1.3.9	get_token()	18
6.1.3.10	get_tokens()	18
6.1.3.11	get_touch_probability(Window w, Touch t)	19
6.1.3.12	get_touches()	19
6.1.3.13	get_window()	20
6.1.3.14	get_windows()	20
6.1.3.15	is_touch_in_key_distribution(Touch touch)	20
6.1.3.16	output_to_csv(String file_name)	21
6.1.3.17	reset()	22
6.1.3.18	set_distribution(Distribution distribution, List< Distribution > key_distribution)	22
6.1.3.19	toString()	22
6.2	runtime.ChainBuilder Class Reference	23
6.2.1	Detailed Description	24
6.2.2	Constructor & Destructor Documentation	24
6.2.2.1	ChainBuilder()	24

6.2.2.2	ChainBuilder(int window, int token, int threshold, int user_model_size, int auth_model_size)	24
6.2.3	Member Function Documentation	24
6.2.3.1	authenticate()	24
6.2.3.2	build_chain_from_csv(File file)	25
6.2.3.3	get_authenticate_state()	25
6.2.3.4	get_authenticate_thread()	26
6.2.3.5	handle_touch(Touch touch)	26
6.2.3.6	parse_csv(File file)	27
6.3	runtime.CompareChains Class Reference	27
6.3.1	Detailed Description	28
6.3.2	Constructor & Destructor Documentation	28
6.3.2.1	CompareChains(Chain user_chain, Chain auth_chain)	28
6.3.3	Member Function Documentation	29
6.3.3.1	get_auth_complete()	29
6.3.3.2	get_auth_probability()	29
6.3.3.3	get_auth_result()	29
6.3.3.4	run()	30
6.3.4	Member Data Documentation	30
6.3.4.1	auth_chain	30
6.3.4.2	authentication_probability	30
6.3.4.3	complete	30
6.3.4.4	is_authentic	30
6.3.4.5	user_chain	30
6.4	rank.CompareChainsRank Class Reference	31
6.4.1	Detailed Description	32
6.4.2	Constructor & Destructor Documentation	32
6.4.2.1	CompareChainsRank(Chain user_chain, Chain auth_chain)	32
6.4.3	Member Function Documentation	32
6.4.3.1	run()	32
6.5	runtime.ChainBuilder.CompareMethod Enum Reference	33

6.5.1	Member Data Documentation	33
6.5.1.1	PROBABILITY_VECTOR_DIFFERENCE	33
6.6	rank.CompleteProbability Class Reference	33
6.6.1	Detailed Description	33
6.6.2	Constructor & Destructor Documentation	34
6.6.2.1	CompleteProbability(Chain chain)	34
6.6.3	Member Function Documentation	34
6.6.3.1	compute_probability()	34
6.7	runtime.Operation_thread.Computation Enum Reference	35
6.7.1	Member Data Documentation	35
6.7.1.1	DISTRIBUTION	35
6.7.1.2	KEY_DISTRIBUTION	35
6.7.1.3	PROBABILITY	35
6.7.1.4	TOKEN	35
6.7.1.5	WINDOW	35
6.8	test.Main.TestFiles.Concentration Enum Reference	35
6.8.1	Constructor & Destructor Documentation	36
6.8.1.1	Concentration(String description, int identifier, double value)	36
6.8.2	Member Function Documentation	36
6.8.2.1	get_identifier()	36
6.8.2.2	get_value()	36
6.8.2.3	toString()	36
6.8.3	Member Data Documentation	36
6.8.3.1	HIGH	36
6.8.3.2	LOW	36
6.8.3.3	MEDIUM	36
6.9	test.Main.TestFiles.Distribution Enum Reference	36
6.9.1	Constructor & Destructor Documentation	37
6.9.1.1	Distribution(String description, int identifier, double value)	37
6.9.2	Member Function Documentation	37

6.9.2.1	get_identifier()	37
6.9.2.2	get_value()	37
6.9.2.3	toString()	37
6.9.3	Member Data Documentation	37
6.9.3.1	ABNORMAL	37
6.9.3.2	NORMAL	37
6.9.3.3	RANDOM	37
6.10	components.Distribution Class Reference	37
6.10.1	Detailed Description	38
6.10.2	Constructor & Destructor Documentation	38
6.10.2.1	Distribution(List< Touch > touches)	38
6.10.2.2	Distribution(List< Touch > touches, int keycode)	38
6.10.2.3	Distribution(Distribution d)	38
6.10.3	Member Function Documentation	38
6.10.3.1	equals(Object o)	38
6.10.3.2	get_average()	39
6.10.3.3	get_keycode()	39
6.10.3.4	get_max()	40
6.10.3.5	get_min()	40
6.10.3.6	get_standard_deviation()	40
6.10.3.7	update(List< Touch > touches)	41
6.11	test.Main Class Reference	41
6.11.1	Detailed Description	41
6.11.2	Member Function Documentation	42
6.11.2.1	main(String args[])	42
6.12	gui.Marcov_console_panel Class Reference	43
6.12.1	Detailed Description	43
6.12.2	Constructor & Destructor Documentation	44
6.12.2.1	Marcov_console_panel()	44
6.13	gui.Marcov_file_display_panel Class Reference	44

6.13.1 Detailed Description	45
6.13.2 Constructor & Destructor Documentation	45
6.13.2.1 <code>Marcov_file_display_panel()</code>	45
6.14 <code>gui.Marcov_frame</code> Class Reference	45
6.14.1 Detailed Description	46
6.14.2 Constructor & Destructor Documentation	46
6.14.2.1 <code>Marcov_frame()</code>	46
6.14.3 Member Function Documentation	46
6.14.3.1 <code>close()</code>	46
6.15 <code>gui.Marcov_options_panel</code> Class Reference	46
6.15.1 Constructor & Destructor Documentation	47
6.15.1.1 <code>Marcov_options_panel()</code>	48
6.16 <code>data_analysis.Model_compare</code> Class Reference	48
6.16.1 Detailed Description	49
6.16.2 Member Function Documentation	49
6.16.2.1 <code>main(String[] args)</code>	49
6.17 <code>data_analysis.Model_compare_thread</code> Class Reference	50
6.17.1 Detailed Description	51
6.17.2 Constructor & Destructor Documentation	51
6.17.2.1 <code>Model_compare_thread(String base_data_path, String auth_data_path, int base_model_size, int auth_model_size, int window_size, int token_size, int threshold)</code>	51
6.17.3 Member Function Documentation	51
6.17.3.1 <code>get_auth_data_path()</code>	51
6.17.3.2 <code>get_auth_model_size()</code>	51
6.17.3.3 <code>get_auth_probability_list()</code>	51
6.17.3.4 <code>get_base_data_path()</code>	51
6.17.3.5 <code>get_base_model_size()</code>	51
6.17.3.6 <code>get_threshold()</code>	51
6.17.3.7 <code>get_token_size()</code>	51
6.17.3.8 <code>get_window_size()</code>	51

6.17.3.9	run()	52
6.17.4	Member Data Documentation	52
6.17.4.1	average_authentication_probability	52
6.17.4.2	max_authentication_probability	52
6.17.4.3	min_authentication_probability	52
6.18	runtime.Operation_thread Class Reference	52
6.18.1	Detailed Description	53
6.18.2	Constructor & Destructor Documentation	54
6.18.2.1	Operation_thread(Chain chain, Computation computation)	54
6.18.3	Member Function Documentation	54
6.18.3.1	run()	54
6.19	test.Main.TestFiles.PressureAmount Enum Reference	54
6.19.1	Constructor & Destructor Documentation	55
6.19.1.1	PressureAmount(String description, int identifier, double value)	55
6.19.2	Member Function Documentation	55
6.19.2.1	get_identifier()	55
6.19.2.2	get_value()	55
6.19.2.3	toString()	55
6.19.3	Member Data Documentation	55
6.19.3.1	HIGH	55
6.19.3.2	LOW	55
6.19.3.3	MEDIUM	55
6.20	test.Print_model Class Reference	55
6.20.1	Detailed Description	55
6.20.2	Member Function Documentation	56
6.20.2.1	main(String[] args)	56
6.21	gui.StartGUI Class Reference	56
6.21.1	Detailed Description	57
6.21.2	Member Function Documentation	57
6.21.2.1	exit()	57

6.21.2.2	<code>main(String[] args)</code>	58
6.22	<code>runtime.ChainBuilder.State</code> Enum Reference	58
6.22.1	Member Data Documentation	58
6.22.1.1	<code>IN_PROGRESS</code>	58
6.22.1.2	<code>SUCCESS</code>	58
6.23	<code>data_analysis.Statistics</code> Class Reference	58
6.23.1	Detailed Description	59
6.23.2	Member Function Documentation	59
6.23.2.1	<code>authentication_accuracy(double authentication_percentage, List< Double > should_authenticate_percentages, List< Double > should_not_authenticate_percentages)</code>	59
6.23.2.2	<code>best_authentication_percentage(List< Double > should_authenticate_percentages, List< Double > should_not_authenticate_percentages)</code>	59
6.23.2.3	<code>equal_false_positive_negative_authentication_percentage(List< Double > should_authenticate_percentages, List< Double > should_not_authenticate percentages)</code>	60
6.23.2.4	<code>false_negative_percentage(double authentication_percentage, List< Double > should_authenticate_percentages, List< Double > should_not_authenticate percentages)</code>	60
6.23.2.5	<code>false_positive_percentage(double authentication_percentage, List< Double > should_authenticate_percentages, List< Double > should_not_authenticate percentages)</code>	61
6.23.2.6	<code>main(String args[])</code>	61
6.23.2.7	<code>minimize_false_positive_authentication_percentage(List< Double > should_authenticate_percentages, List< Double > should_not_authenticate percentages)</code>	61
6.24	<code>components.Token</code> Class Reference	62
6.24.1	Detailed Description	63
6.24.2	Constructor & Destructor Documentation	63
6.24.2.1	<code>Token(Distribution distribution, int total_tokens, int token_index, double standard_deviations, Type type)</code>	63
6.24.2.2	<code>Token(Distribution distribution, int total_tokens, int token_index, Type type)</code>	64
6.24.2.3	<code>Token(double range_min, double range_max, int total_tokens, int token_index, Type type)</code>	64
6.24.3	Member Function Documentation	64
6.24.3.1	<code>contains(Touch touch)</code>	64
6.24.3.2	<code>equals(Object o_t)</code>	65

6.24.3.3	get_acceptable_wildcards(int total_items)	65
6.24.3.4	get_max()	65
6.24.3.5	get_min()	66
6.24.3.6	get_total_wildcards()	66
6.24.3.7	increment_high_wildcards()	66
6.24.3.8	increment_low_wildcards()	66
6.24.3.9	is_high_wildcard(Touch touch)	66
6.24.3.10	is_low_wildcard(Touch touch)	67
6.25	components.Touch Class Reference	67
6.25.1	Detailed Description	68
6.25.2	Constructor & Destructor Documentation	68
6.25.2.1	Touch(int keycode, double pressure, long timestamp)	68
6.25.2.2	Touch(Touch t)	68
6.25.3	Member Function Documentation	68
6.25.3.1	compare_with_token(List< Token > tokens, Touch other_touch)	68
6.25.3.2	compareTo(Touch other_touch)	69
6.25.3.3	get_key()	69
6.25.3.4	get_pressure()	70
6.25.3.5	get_probability(Window preceeding_window)	70
6.25.3.6	get_timestamp()	70
6.25.3.7	hashCode()	70
6.25.3.8	set_probability(Window preceeding_window, double p)	71
6.25.3.9	toString()	71
6.26	trie.Trie Class Reference	71
6.26.1	Detailed Description	72
6.26.2	Constructor & Destructor Documentation	72
6.26.2.1	Trie()	72
6.26.2.2	Trie(Trie t)	72
6.26.3	Member Function Documentation	72
6.26.3.1	clear()	72

6.26.3.2	get_index_list(String s)	72
6.26.3.3	insertString(String s, int index)	73
6.26.3.4	occurrence_count(String s)	73
6.26.3.5	printSorted(TrieNode node, String s)	73
6.27	trie.TrieList Class Reference	73
6.27.1	Detailed Description	75
6.27.2	Constructor & Destructor Documentation	75
6.27.2.1	TrieList()	75
6.27.2.2	TrieList(TrieList t)	75
6.27.3	Member Function Documentation	75
6.27.3.1	add(Window arg0)	75
6.27.3.2	add(int arg0, Window arg1)	75
6.27.3.3	addAll(Collection<?extends Window > arg0)	75
6.27.3.4	addAll(int arg0, Collection<?extends Window > arg1)	76
6.27.3.5	clear()	76
6.27.3.6	occurrence_count(Window w)	76
6.27.3.7	remove(Object arg0)	77
6.27.3.8	remove(int arg0)	77
6.27.3.9	removeAll(Collection<?> arg0)	77
6.27.3.10	retainAll(Collection<?> arg0)	77
6.27.3.11	set(int arg0, Window arg1)	78
6.27.3.12	set_tokens(List< Token > tokens)	78
6.27.3.13	successor_count(List< Touch > successor_list, Window window, Touch touch)	78
6.28	components.Token.Type Enum Reference	79
6.28.1	Detailed Description	79
6.28.2	Member Data Documentation	79
6.28.2.1	combined	79
6.28.2.2	keycode_mu	79
6.28.2.3	linear	79
6.29	junit.Unit_CompareChainsRank Class Reference	79

6.29.1 Detailed Description	79
6.29.2 Member Function Documentation	80
6.29.2.1 init()	80
6.29.2.2 test_authentication_probability()	80
6.30 junit.Unit_CompleteProbability Class Reference	80
6.30.1 Detailed Description	81
6.30.2 Member Function Documentation	81
6.30.2.1 init()	81
6.30.2.2 test_replica_distribution()	81
6.31 test.UnitCompareChainsRank Class Reference	82
6.31.1 Detailed Description	82
6.31.2 Member Function Documentation	83
6.31.2.1 init()	83
6.31.2.2 test()	83
6.31.2.3 test_chain_to_graph()	83
6.31.2.4 test_touch_index()	83
6.31.2.5 test_touch_window()	83
6.32 test.UnitRankCompare Class Reference	84
6.32.1 Detailed Description	84
6.32.2 Member Function Documentation	85
6.32.2.1 init()	85
6.32.2.2 test()	85
6.32.2.3 test_auth_probability()	85
6.32.2.4 test_compare_correct()	85
6.33 components.Window Class Reference	86
6.33.1 Detailed Description	87
6.33.2 Constructor & Destructor Documentation	87
6.33.2.1 Window(List< Touch > touches)	87
6.33.2.2 Window(Window w)	87
6.33.3 Member Function Documentation	87
6.33.3.1 compare_with_token(List< Token > tokens, Window other_window)	87
6.33.3.2 compareTo(Window other_window)	88
6.33.3.3 get_touch_list()	88
6.33.3.4 hashCode()	88
6.33.3.5 size()	88
6.33.3.6 toString()	88

7	File Documentation	89
7.1	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/Chain.java File Reference	89
7.2	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/Distribution.java File Reference	89
7.3	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/Token.java File Reference	89
7.4	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/Touch.java File Reference	90
7.5	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/Window.java File Reference	90
7.6	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/data_analysis/Model_↔compare.java File Reference	90
7.7	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/data_analysis/Model_↔compare_thread.java File Reference	91
7.8	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/data_analysis/Statistics.java File Reference	91
7.9	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/Marcov_console_panel.java File Reference	91
7.10	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/Marcov_file_display_↔panel.java File Reference	91
7.11	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/Marcov_frame.java File Reference	92
7.12	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/Marcov_options_panel.java File Reference	92
7.13	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/StartGUI.java File Reference	92
7.14	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/junit/Unit_CompareChains↔Rank.java File Reference	93
7.15	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/junit/Unit_CompleteProbability.java File Reference	93
7.16	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/rank/CompareChainsRank.java File Reference	93
7.17	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/rank/CompleteProbability.java File Reference	93
7.18	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/runtime/ChainBuilder.java File Reference	94
7.19	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/runtime/CompareChains.java File Reference	94

7.20	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/runtime/Operation_thread.java File Reference	94
7.21	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/Main.java File Reference .	95
7.22	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/Print_model.java File Ref- erence	95
7.23	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/UnitCompareChains↵ Rank.java File Reference	95
7.24	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/UnitRankCompare.java File Reference	96
7.25	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/Utilities.java File Reference	96
7.26	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/trie/Trie.java File Reference . .	96
7.27	/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/trie/TrieList.java File Reference	96
Index		97

Chapter 1

Namespace Index

1.1 Packages

Here are the packages with brief descriptions (if available):

components	9
data_analysis	9
gui	10
junit	10
rank	10
runtime	10
test	11
trie	11

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

components.Chain	13
runtime.ChainBuilder	23
Comparable	
components.Touch	67
components.Window	86
runtime.ChainBuilder.CompareMethod	33
rank.CompleteProbability	33
runtime.Operation_thread.Computation	35
test.Main.TestFiles.Concentration	35
test.Main.TestFiles.Distribution	36
components.Distribution	37
test.Main	41
data_analysis.Model_compare	48
test.Main.TestFiles.PressureAmount	54
test.Print_model	55
Runnable	
data_analysis.Model_compare_thread	50
runtime.CompareChains	27
rank.CompareChainsRank	31
runtime.Operation_thread	52
gui.StartGUI	56
runtime.ChainBuilder.State	58
data_analysis.Statistics	58
components.Token	62
trie.Trie	71
components.Token.Type	79
junit.Unit_CompareChainsRank	79
junit.Unit_CompleteProbability	80
test.UnitCompareChainsRank	82
test.UnitRankCompare	84
ArrayList	
trie.TrieList	73
JFrame	
gui.Marcov_frame	45
JPanel	
gui.Marcov_console_panel	43
gui.Marcov_file_display_panel	44
gui.Marcov_options_panel	46

Chapter 3

Class Index

3.1 Class List

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

components.Chain	
Markov Chain built using keyboard tokens	13
runtime.ChainBuilder	
Wrapper around Chain used to make using the model easier in a real application	23
runtime.CompareChains	
Use the compare method of the Chain class to determine an authentication probability between 0 and 1	27
rank.CompareChainsRank	
Use PageRank algorithm (library) to compare chains	31
runtime.ChainBuilder.CompareMethod	33
rank.CompleteProbability	
This class computes probability in a different way from what is contained in the Chain class	33
runtime.Operation_thread.Computation	35
test.Main.TestFiles.Concentration	35
test.Main.TestFiles.Distribution	36
components.Distribution	
Used to compute and store max, min, std_deviation, and average for a list of Touch	37
test.Main	
This class is used to test that the model is being built correctly	41
gui.Marcov_console_panel	
Displays messages printed to stdout	43
gui.Marcov_file_display_panel	
Displays files relevant to test code	44
gui.Marcov_frame	
Display frame to contain buttons for running test code and panels to view results	45
gui.Marcov_options_panel	46
data_analysis.Model_compare	
Analysis class used to compare and analyze data gathered from users	48
data_analysis.Model_compare_thread	
Compare Markov Chains on their own thread	50
runtime.Operation_thread	
UNUSED	52
test.Main.TestFiles.PressureAmount	54
test.Print_model	
This class will print out the model constructed from the designated file	55

gui.StartGUI	
GUI useful for testing	56
runtime.ChainBuilder.State	58
data_analysis.Statistics	
Generates statistics on results generated by model_compare.java	58
components.Token	
This class represents a token within the model	62
components.Touch	
This class represents a touch event	67
trie.Trie	
Implementation of Prefix Tree	71
trie.TrieList	
Wrapper around Trie used to maintain an ordering among the stored elements	73
components.Token.Type	
Specify the type of token we want to build	79
junit.Unit_CompareChainsRank	
Goal is to test compare chains rank functionality	79
junit.Unit_CompleteProbability	
Unit test demonstrating how to compute probability	80
test.UnitCompareChainsRank	
JUnit test for testing PageRank version of compairason Not that the PageRank implementation is not currently functional	82
test.UnitRankCompare	
Test the compairason with ranks	84
components.Window	
This class will store and provide functions for a single window within the model	86

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/Chain.java	89
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/Distribution.java	89
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/Token.java	89
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/Touch.java	90
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/Window.java	90
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/data_analysis/Model_compare.java	90
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/data_analysis/Model_compare_↵ thread.java	91
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/data_analysis/Statistics.java	91
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/Marcov_console_panel.java . . .	91
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/Marcov_file_display_panel.java .	91
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/Marcov_frame.java	92
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/Marcov_options_panel.java . . .	92
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/StartGUI.java	92
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/junit/Unit_CompareChainsRank.java	93
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/junit/Unit_CompleteProbability.java .	93
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/rank/CompareChainsRank.java . . .	93
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/rank/CompleteProbability.java . . .	93
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/runtime/ChainBuilder.java	94
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/runtime/CompareChains.java	94
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/runtime/Operation_thread.java . . .	94
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/Main.java	95
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/Print_model.java	95
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/UnitCompareChainsRank.java .	95
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/UnitRankCompare.java	96
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/Utilities.java	96
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/trie/Trie.java	96
/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/trie/TrieList.java	96

Chapter 5

Namespace Documentation

5.1 Package components

Classes

- class [Chain](#)
Markov [Chain](#) built using keyboard tokens.
- class [Distribution](#)
Used to compute and store max, min, std_deviation, and average for a list of [Touch](#).
- class [Token](#)
This class represents a token within the model.
- class [Touch](#)
This class represents a touch event.
- class [Window](#)
This class will store and provide functions for a single window within the model.

5.2 Package data_analysis

Classes

- class [Model_compare](#)
Analysis class used to compare and analyze data gathered from users.
- class [Model_compare_thread](#)
Compare Markov Chains on their own thread.
- class [Statistics](#)
Generates statistics on results generated by model_compare.java.

5.3 Package gui

Classes

- class [Marcov_console_panel](#)
Displays messages printed to stdout.
- class [Marcov_file_display_panel](#)
Displays files relevant to test code.
- class [Marcov_frame](#)
Display frame to contain buttons for running test code and panels to view results.
- class [Marcov_options_panel](#)
- class [StartGUI](#)
GUI useful for testing.

5.4 Package junit

Classes

- class [Unit_CompareChainsRank](#)
goal is to test compare chains rank functionality
- class [Unit_CompleteProbability](#)
unit test demonstrating how to compute probability

5.5 Package rank

Classes

- class [CompareChainsRank](#)
Use PageRank algorithm (library) to compare chains.
- class [CompleteProbability](#)
This class computes probability in a different way from what is contained in the Chain class.

5.6 Package runtime

Classes

- class [ChainBuilder](#)
Wrapper around Chain used to make using the model easier in a real application.
- class [CompareChains](#)
Use the compare method of the Chain class to determine an authentication probability between 0 and 1.
- class [Operation_thread](#)
UNUSED.

5.7 Package test

Classes

- class [Main](#)
This class is used to test that the model is being built correctly.
- class [Print_model](#)
This class will print out the model constructed from the designated file.
- class [UnitCompareChainsRank](#)
JUnit test for testing PageRank version of comparison Not that the PageRank implementation is not currently functional.
- class [UnitRankCompare](#)
Test the comparison with ranks.

5.8 Package trie

Classes

- class [Trie](#)
Implementation of Prefix Tree.
- class [TrieList](#)
Wrapper around [Trie](#) used to maintain an ordering among the stored elements.

Chapter 6

Class Documentation

6.1 components.Chain Class Reference

Markov [Chain](#) built using keyboard tokens.

Public Member Functions

- [Chain](#) (int window, int token, int threshold, int model_size)
- [Chain](#) ([Chain](#) c)
copy constructor. New chain object should have the same state as the old with different object references.
- void [add_touch](#) ([Touch](#) touch)
- void [add_touch_list](#) (List< [Touch](#) > t)
- void [set_distribution](#) ([Distribution](#) distribution, List< [Distribution](#) > key_distribution)
allows distribution to be set.
- double [get_touch_probability](#) ([Window](#) w, [Touch](#) t)
returns the probability of a given touch (at the i'th index) based on the model. This will depend on the preceding touches, in [Window](#). A request for one probability will necessarily result in all of the probabilities being computed.
- [Distribution](#) [get_distribution](#) ()
returns the distribution of the data as a whole
- List< [Distribution](#) > [get_key_distribution](#) ()
returns a list of distributions for each key
- int [get_window](#) ()
- int [get_token](#) ()
- int [get_model_size](#) ()
- int [get_threshold](#) ()
- void [reset](#) ()
resets the object.. this is the same as constructing a new chain, but faster
- void [compute_uncomputed](#) ()
computes all uncomputed aspects of the chain
- double [compare_to](#) ([Chain](#) auth_chain)
returns the percent difference between this chain and auth_chain.
- boolean [is_touch_in_key_distribution](#) ([Touch](#) touch)
returns true if a touch is within 2 sigma for it's key distribution
- List< [Window](#) > [get_windows](#) ()
handle requests for windows

- List< [Token](#) > [get_tokens](#) ()
handle requests for tokens
- List< [Touch](#) > [get_touches](#) ()
get a list of all touches in the chain
- String [toString](#) ()
prints out all of the touches in order
- void [output_to_csv](#) (String file_name)
NOT USEFUL IN ANDROID. This is used for debugging purposes. Outputs the model to a csv file in a readable format.

6.1.1 Detailed Description

Markov [Chain](#) built using keyboard tokens.

This class was designed to be used with the keyboard of a mobile phone. The soft keyboard of this device produces (key, pressure) values. These (key, pressure) values become the tokens in our Markov [Chain](#).

6.1.2 Constructor & Destructor Documentation

6.1.2.1 `components.Chain.Chain (int window, int token, int threshold, int model_size)`

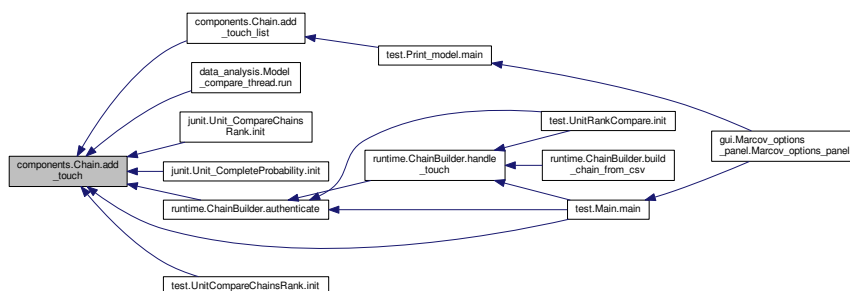
6.1.2.2 `components.Chain.Chain (Chain c)`

copy constructor. New chain object should have the same state as the old with differant object references.

6.1.3 Member Function Documentation

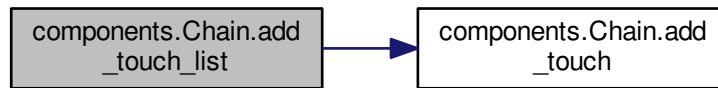
6.1.3.1 `void components.Chain.add_touch (Touch touch)`

Here is the caller graph for this function:

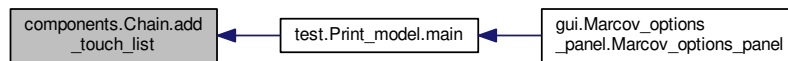


6.1.3.2 void components.Chain.add_touch_list (List< Touch > t)

Here is the call graph for this function:



Here is the caller graph for this function:

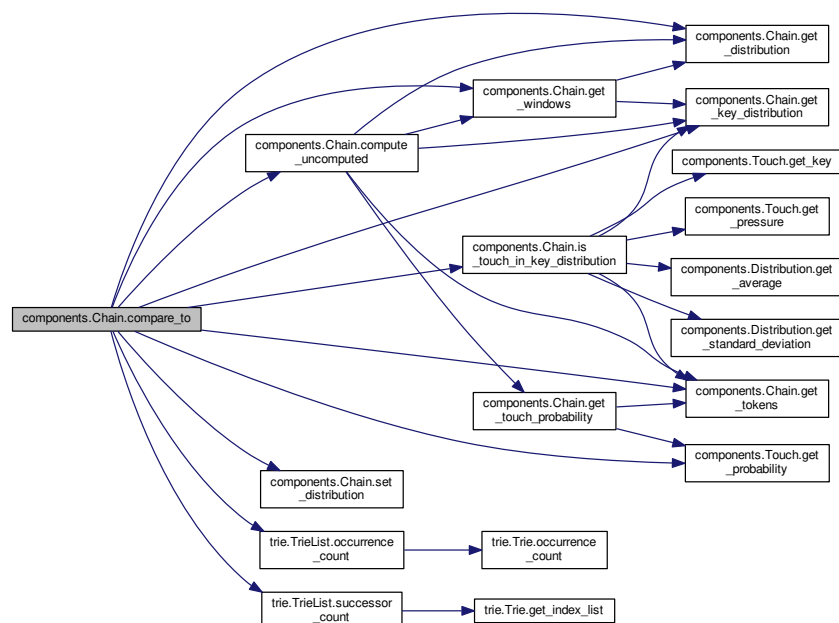


6.1.3.3 double components.Chain.compare_to (Chain auth_chain)

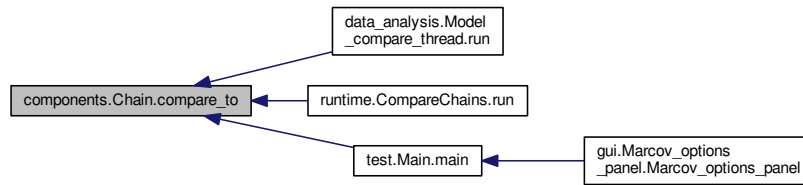
returns the percent difference between this chain and `auth_chain`.

the value returned will be between 0 and 1 0 indicates there is no difference 1 indicates there is a large difference

Here is the call graph for this function:



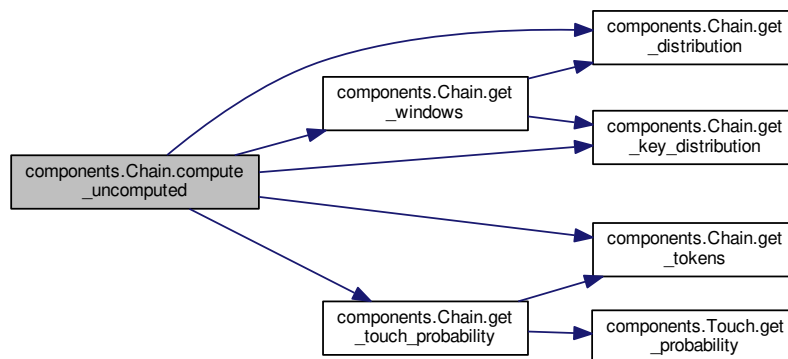
Here is the caller graph for this function:



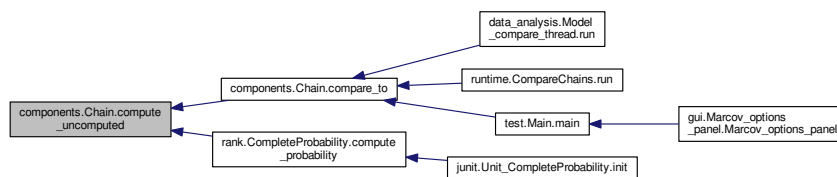
6.1.3.4 void components.Chain.compute_uncomputed ()

computes all uncomputed aspects of the chain

Here is the call graph for this function:



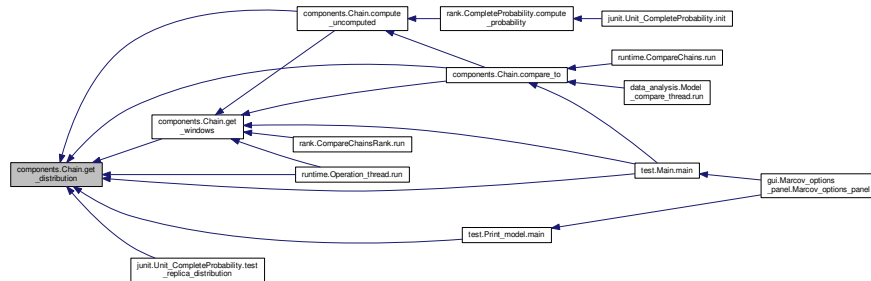
Here is the caller graph for this function:



6.1.3.5 Distribution components.Chain.get_distribution ()

returns the distribution of the data as a whole

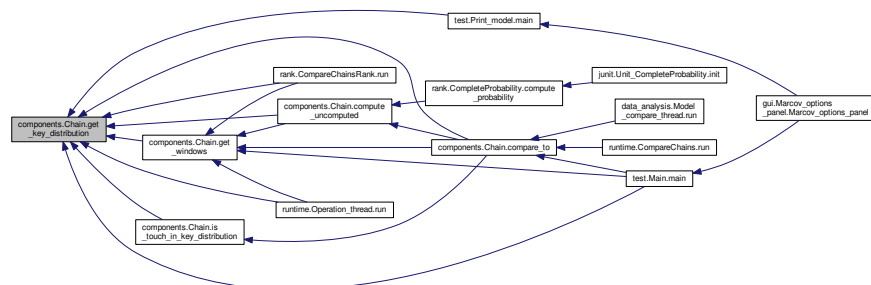
Here is the caller graph for this function:



6.1.3.6 List<Distribution> components.Chain.get_key_distribution ()

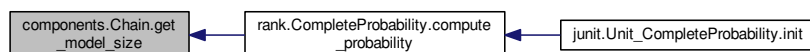
returns a list of distributions for each key

Here is the caller graph for this function:



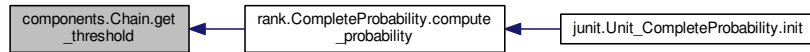
6.1.3.7 int components.Chain.get_model_size ()

Here is the caller graph for this function:



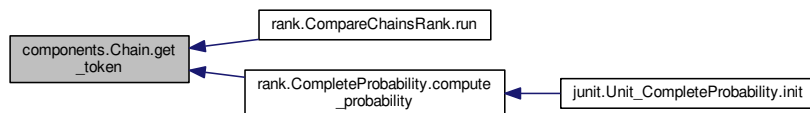
6.1.3.8 int components.Chain.get_threshold ()

Here is the caller graph for this function:



6.1.3.9 int components.Chain.get_token ()

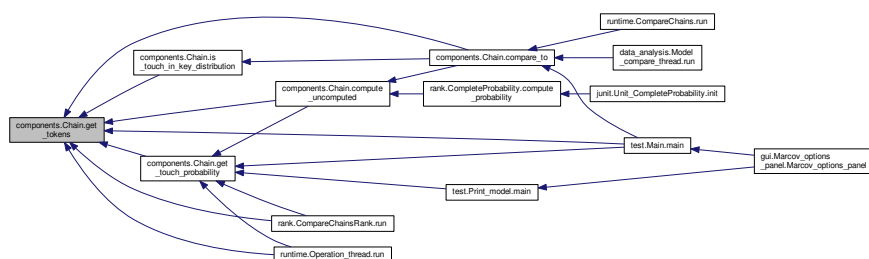
Here is the caller graph for this function:



6.1.3.10 List<Token> components.Chain.get_tokens ()

handle requests for tokens

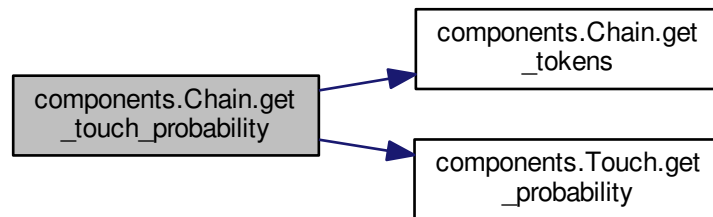
Here is the caller graph for this function:



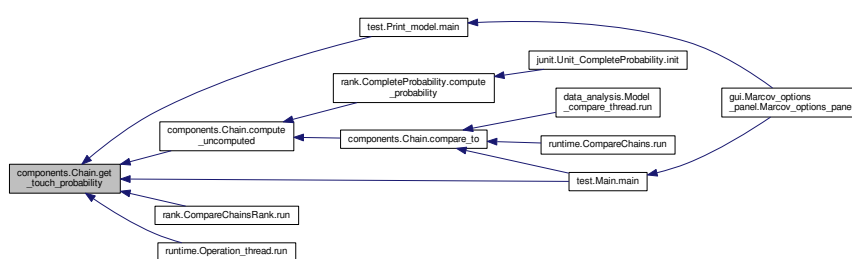
6.1.3.11 double components.Chain.get_touch_probability (Window w, Touch t)

returns the probability of a given touch (at the i'th index) based on the model. This will depend on the preceeding touches, in [Window](#). A request for one probability will necessarily result in all of the probabilities being computed.

Here is the call graph for this function:



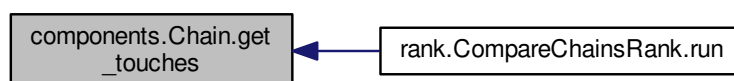
Here is the caller graph for this function:



6.1.3.12 List<Touch> components.Chain.get_touches ()

get a list of all touches in the chain

Here is the caller graph for this function:

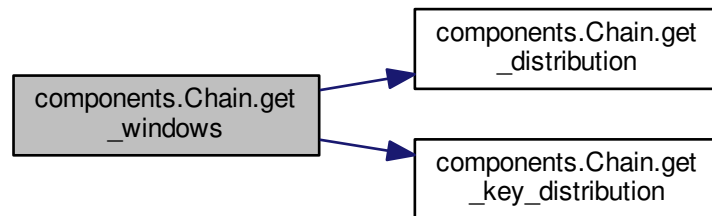


6.1.3.13 `int components.Chain.get_window ()`

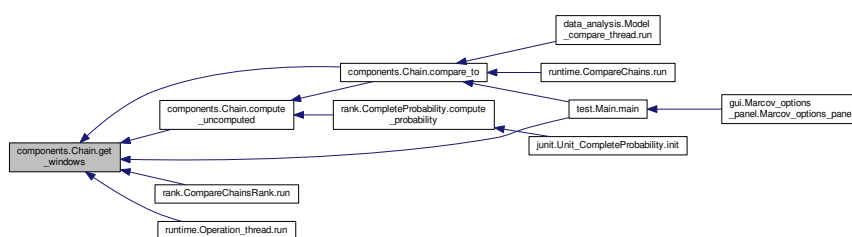
6.1.3.14 `List<Window> components.Chain.get_windows ()`

handle requests for windows

Here is the call graph for this function:



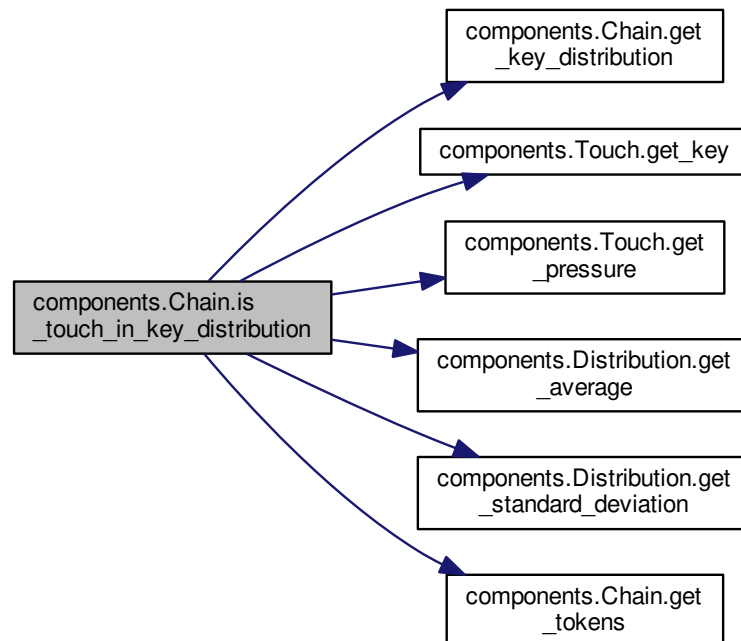
Here is the caller graph for this function:



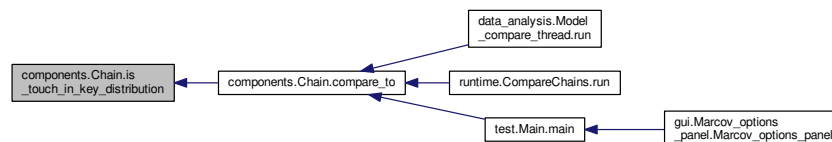
6.1.3.15 `boolean components.Chain.is_touch_in_key_distribution (Touch touch)`

returns true if a touch is within 2 sigma for it's key distribution

Here is the call graph for this function:



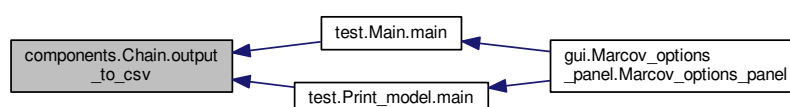
Here is the caller graph for this function:



6.1.3.16 void components.Chain.output_to_csv (String file_name)

NOT USEFUL IN ANDROID. This is used for debugging purposes. Outputs the model to a csv file in a readable format.

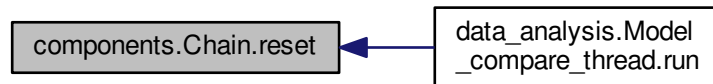
Here is the caller graph for this function:



6.1.3.17 void components.Chain.reset ()

resets the object.. this is the same as constructing a new chain, but faster

Here is the caller graph for this function:

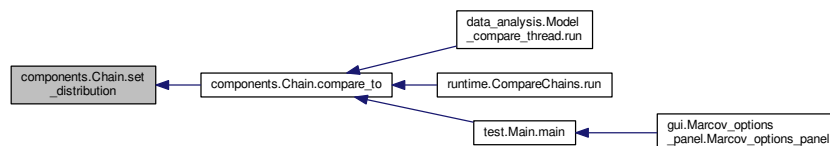


6.1.3.18 void components.Chain.set_distribution (Distribution distribution, List< Distribution > key_distribution)

allows distribution to be set.

If no distribution is set, the distribution for this chain of touches is computed. NOTE the distribution is not maintained when new touches are added.

Here is the caller graph for this function:



6.1.3.19 String components.Chain.toString ()

prints out all of the touches in order

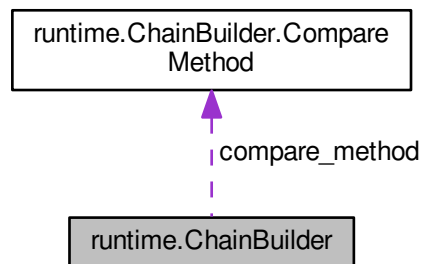
The documentation for this class was generated from the following file:

- `/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/Chain.java`

6.2 runtime.ChainBuilder Class Reference

Wrapper around Chain used to make using the model easier in a real application.

Collaboration diagram for runtime.ChainBuilder:



Classes

- enum [CompareMethod](#)
- enum [State](#)

Public Member Functions

- [ChainBuilder](#) ()
- [ChainBuilder](#) (int window, int token, int threshold, int user_model_size, int auth_model_size)
allow model size, window, token values to be specified.
- void [handle_touch](#) ([Touch](#) touch)
this method should be called in some way whenever there is a touch event in android.
- void [authenticate](#) ()
allow forced authentication from outside of [ChainBuilder](#).
- [CompareChains](#) [get_authenticate_thread](#) ()
return the thread which is performing the authentication.
- [State](#) [get_authenticate_state](#) ()
handle requests for the current state of the authentication
- void [build_chain_from_csv](#) (File file)
this code will NOT BE USEFULL ON ANDROID.

Static Public Member Functions

- static List< [Touch](#) > [parse_csv](#) (File file)
parse the csv file NOT USEFULL ON ANDROID

6.2.1 Detailed Description

Wrapper around Chain used to make using the model easier in a real application.

This class will construct a model with the given parameters. Calling `authenticate` will cause the newest [`user_model_size`] touches to be compared against [`auth_model_size`] touches which immediately precede the touches used in the `user_model`.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 `runtime.ChainBuilder.ChainBuilder ()`

6.2.2.2 `runtime.ChainBuilder.ChainBuilder (int window, int token, int threshold, int user_model_size, int auth_model_size)`

allow model size, window, token values to be specified.

This is mainly for testing purposes

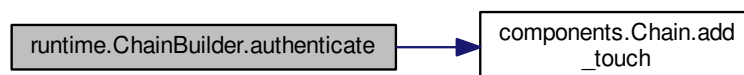
6.2.3 Member Function Documentation

6.2.3.1 `void runtime.ChainBuilder.authenticate ()`

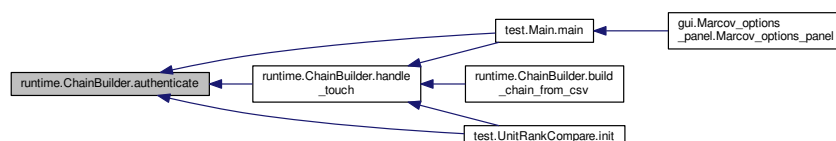
allow forced authentication from outside of [ChainBuilder](#).

this involves starting the [CompareChains](#). this method starts the authentication

Here is the call graph for this function:



Here is the caller graph for this function:

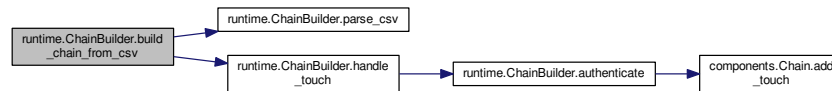


6.2.3.2 void runtime.ChainBuilder.build_chain_from_csv (File file)

this code will NOT BE USEFULL ON ANDROID.

It will build the model from a csv file in the current working directory. It will however utilize the [handle_touch\(\)](#) method to add new touches to the chain. It is simply a matter of where the touches are coming from. TODO move this method to another place. it is only by convience that it exists here now.

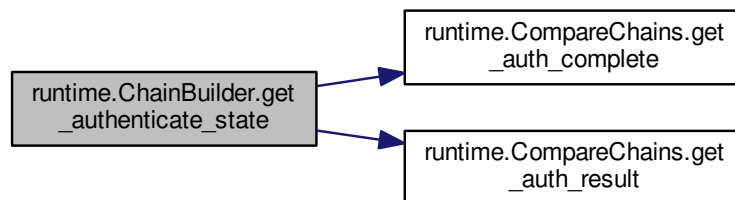
Here is the call graph for this function:



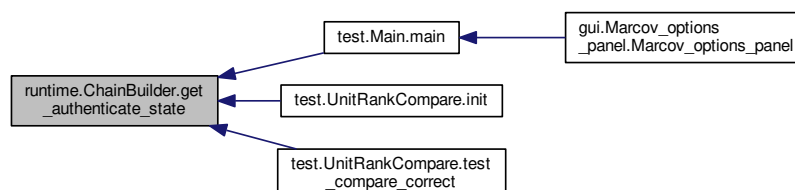
6.2.3.3 State runtime.ChainBuilder.get_authenticate_state ()

handle requests for the current state of the authentication

Here is the call graph for this function:



Here is the caller graph for this function:

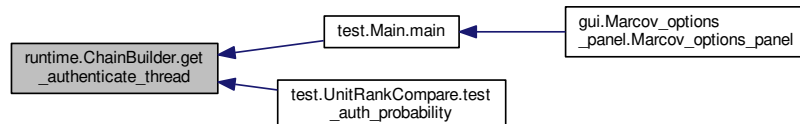


6.2.3.4 CompareChains runtime.ChainBuilder.get_authenticate_thread ()

return the thread which is preforming the authentication.

This method provides no guarentees about the state of the thread. It may even be null!

Here is the caller graph for this function:

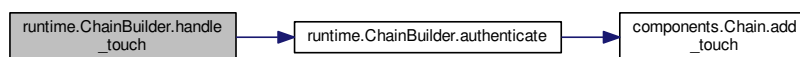


6.2.3.5 void runtime.ChainBuilder.handle_touch (Touch touch)

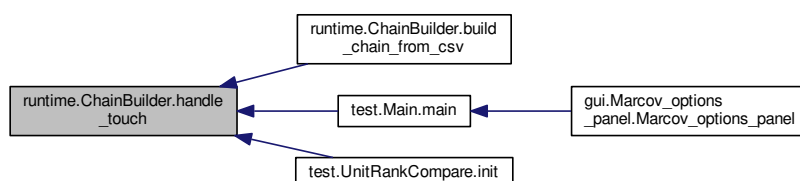
this method should be called in some way whenever there is a touch event in android.

There should be minimal amounts of processing done here so the input to the device doesn't lag. I don't know by what method percicely this will need to be called in the android souce. It could be another class which simply handles touch events, or from the pre-existing android archetecture.

Here is the call graph for this function:



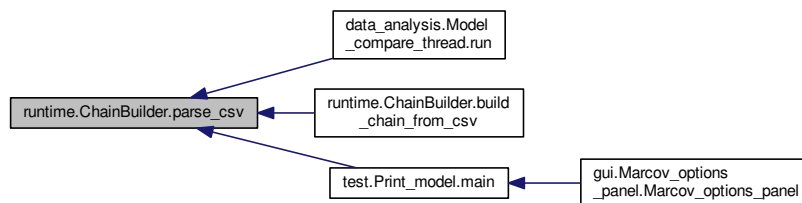
Here is the caller graph for this function:



6.2.3.6 static List<Touch> runtime.ChainBuilder.parse_csv (File file) [static]

parse the csv file NOT USEFULL ON ANDROID

Here is the caller graph for this function:



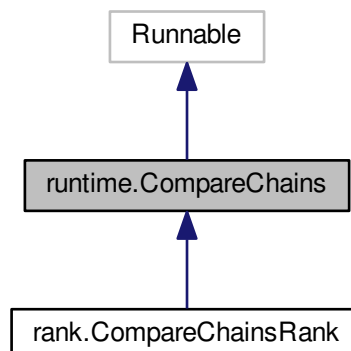
The documentation for this class was generated from the following file:

- `/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/runtime/ChainBuilder.java`

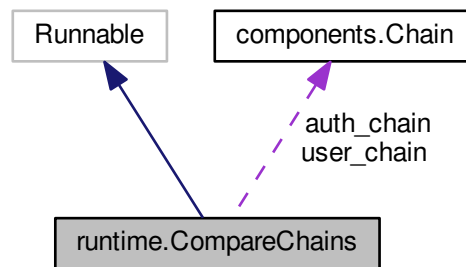
6.3 runtime.CompareChains Class Reference

Use the compare method of the Chain class to determine an authentication probability between 0 and 1.

Inheritance diagram for runtime.CompareChains:



Collaboration diagram for runtime.CompareChains:



Public Member Functions

- [CompareChains](#) ([Chain user_chain](#), [Chain auth_chain](#))
will need to make copies of the chains passed in so they do not get updated by something else during the comparason
- void [run](#) ()
compare user_chain and auth_chain and choose what to do with the result
- double [get_auth_probability](#) ()
returns the probability with which the
- boolean [get_auth_result](#) ()
returns the result of the authentication.
- boolean [get_auth_complete](#) ()

Protected Attributes

- volatile boolean [is_authentic](#)
- volatile boolean [complete](#)
- [Chain user_chain](#)
- [Chain auth_chain](#)
- volatile double [authentication_probability](#)

6.3.1 Detailed Description

Use the compare method of the Chain class to determine an authetnication probability between 0 and 1.

This class was designed to allow for comparing chains to happen on a different thread.

6.3.2 Constructor & Destructor Documentation

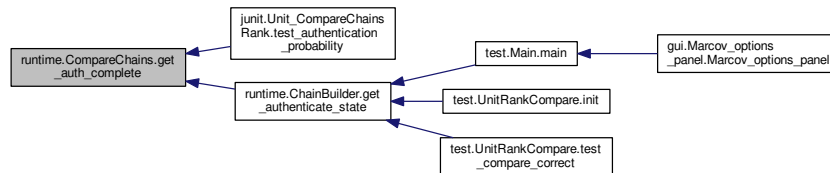
6.3.2.1 runtime.CompareChains.CompareChains ([Chain user_chain](#), [Chain auth_chain](#))

will need to make copies of the chains passed in so they do not get updated by something else during the comparason

6.3.3 Member Function Documentation

6.3.3.1 boolean runtime.CompareChains.get_auth_complete ()

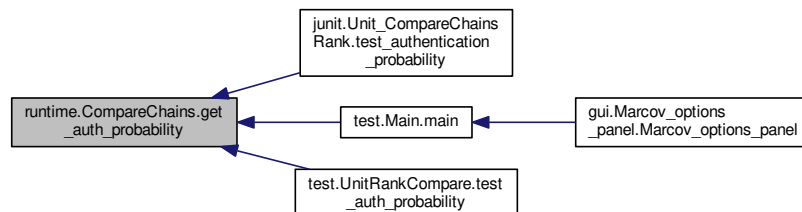
Here is the caller graph for this function:



6.3.3.2 double runtime.CompareChains.get_auth_probability ()

returns the probability with which the

Here is the caller graph for this function:

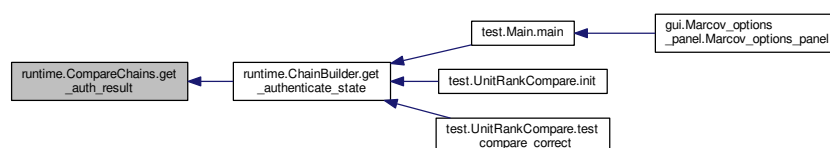


6.3.3.3 boolean runtime.CompareChains.get_auth_result ()

returns the result of the authentication.

This method does not provide any guarentees that the compairason has finsihed yet. If the compairason has not yet finished it will return false;

Here is the caller graph for this function:

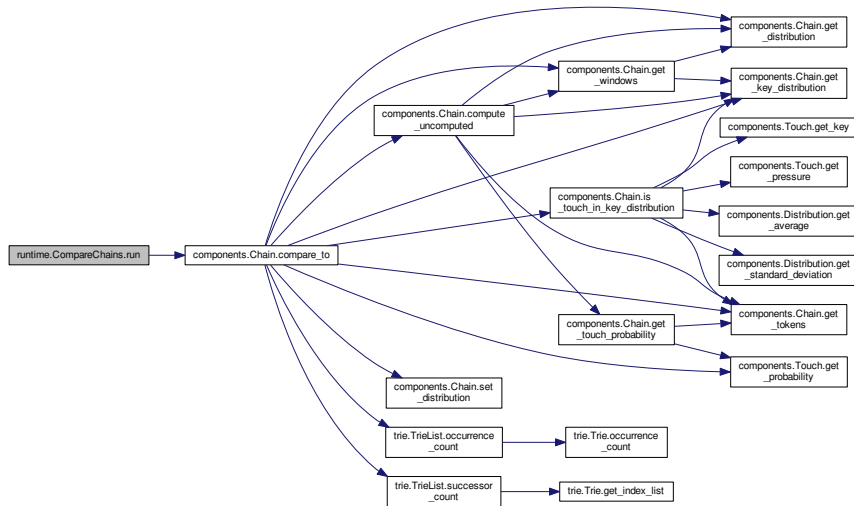


6.3.3.4 void runtime.CompareChains.run ()

compare user_chain and auth_chain and choose what to do with the result

perform the comparison now that the values are cached in the Chain's

Here is the call graph for this function:



6.3.4 Member Data Documentation

6.3.4.1 Chain runtime.CompareChains.auth_chain [protected]

6.3.4.2 volatile double runtime.CompareChains.authentication_probability [protected]

6.3.4.3 volatile boolean runtime.CompareChains.complete [protected]

6.3.4.4 volatile boolean runtime.CompareChains.is_authentic [protected]

6.3.4.5 Chain runtime.CompareChains.user_chain [protected]

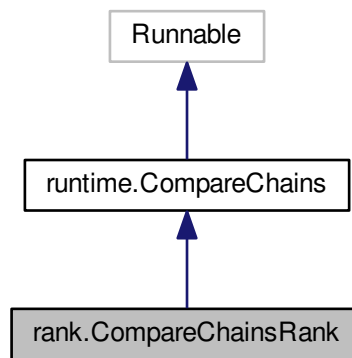
The documentation for this class was generated from the following file:

- /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/runtime/CompareChains.java

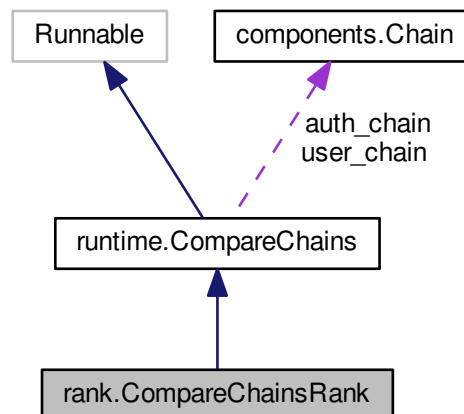
6.4 rank.CompareChainsRank Class Reference

Use PageRank algorithm (library) to compare chains.

Inheritance diagram for rank.CompareChainsRank:



Collaboration diagram for rank.CompareChainsRank:



Public Member Functions

- [CompareChainsRank](#) ([Chain](#) user_chain, [Chain](#) auth_chain)
- void [run](#) ()

overrides the run method to implement the authentication with a page-rank style algorithm.

Additional Inherited Members

6.4.1 Detailed Description

Use PageRank algorithm (library) to compare chains.

Provides an implementation of CompareChains which can be used in place of the chain compairason implemented in the Chain class.

NOTE: this is still a work in progress and does not have very good results yet.

6.4.2 Constructor & Destructor Documentation

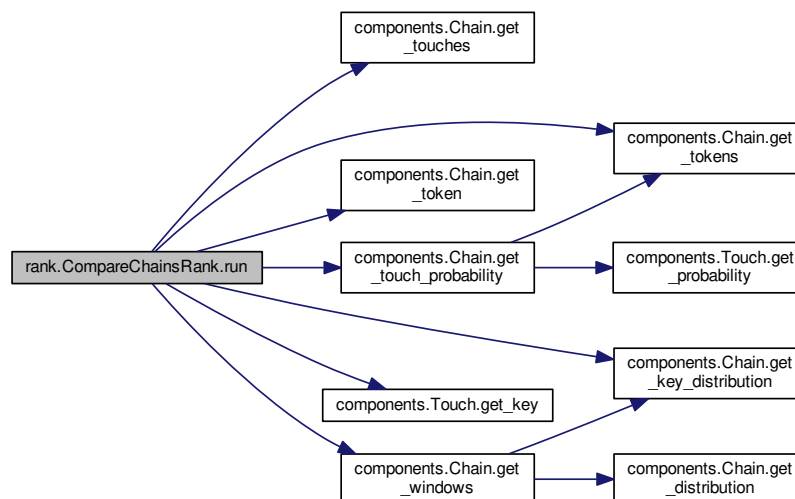
6.4.2.1 `rank.CompareChainsRank.CompareChainsRank (Chain user_chain, Chain auth_chain)`

6.4.3 Member Function Documentation

6.4.3.1 `void rank.CompareChainsRank.run ()`

overrides the run method to implement the authentication with a page-rank style algorithm.

Here is the call graph for this function:



The documentation for this class was generated from the following file:

- `/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/rank/CompareChainsRank.java`

6.5 runtime.ChainBuilder.CompareMethod Enum Reference

Public Attributes

- [PROBABILITY_VECTOR_DIFFERENCE](#)

6.5.1 Member Data Documentation

6.5.1.1 runtime.ChainBuilder.CompareMethod.PROBABILITY_VECTOR_DIFFERENCE

The documentation for this enum was generated from the following file:

- /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/runtime/[ChainBuilder.java](#)

6.6 rank.CompleteProbability Class Reference

This class computes probability in a different way from what is contained in the Chain class.

Public Member Functions

- [CompleteProbability](#) ([Chain](#) chain)
- [Chain compute_probability](#) ()

make a replica of the chain with a window size of 1 and compute the probability.

6.6.1 Detailed Description

This class computes probability in a different way from what is contained in the Chain class.

This class looks at all of the touches to try to determine the probability that from any given touch, it transitions to another.

this is similar to having a window size of 1?

Author

element

6.6.2 Constructor & Destructor Documentation

6.6.2.1 rank.CompleteProbability.CompleteProbability (Chain chain)

6.6.3 Member Function Documentation

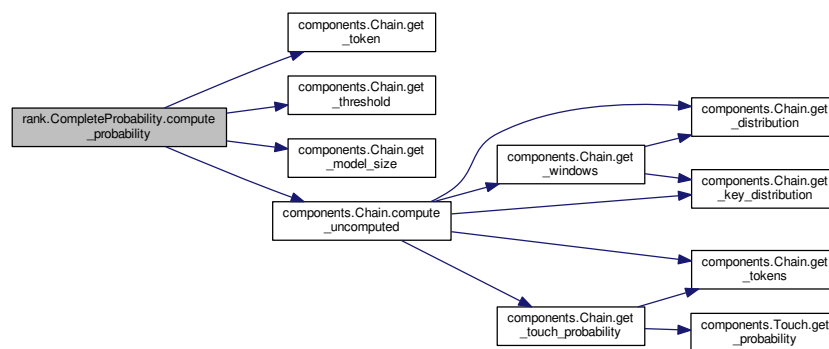
6.6.3.1 Chain rank.CompleteProbability.compute_probability ()

make a replica of the chain with a window size of 1 and compute the probability.

Returns

replica chain

Here is the call graph for this function:



Here is the caller graph for this function:



The documentation for this class was generated from the following file:

- `/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/rank/CompleteProbability.java`

6.7 runtime.Operation_thread.Computation Enum Reference

Public Attributes

- [DISTRIBUTION](#)
- [KEY_DISTRIBUTION](#)
- [WINDOW](#)
- [TOKEN](#)
- [PROBABILITY](#)

6.7.1 Member Data Documentation

6.7.1.1 runtime.Operation_thread.Computation.DISTRIBUTION

6.7.1.2 runtime.Operation_thread.Computation.KEY_DISTRIBUTION

6.7.1.3 runtime.Operation_thread.Computation.PROBABILITY

6.7.1.4 runtime.Operation_thread.Computation.TOKEN

6.7.1.5 runtime.Operation_thread.Computation.WINDOW

The documentation for this enum was generated from the following file:

- [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/runtime/Operation_thread.java](#)

6.8 test.Main.TestFiles.Concentration Enum Reference

Public Member Functions

- [Concentration](#) (String description, int identifier, double value)
- String [toString](#) ()
- int [get_identifier](#) ()
- double [get_value](#) ()

Public Attributes

- [HIGH](#) =("High, [std deviation]", 0, 0)
- [MEDIUM](#) =("Medium, [std deviation]", 1, 0)
- [LOW](#)

6.8.1 Constructor & Destructor Documentation

6.8.1.1 `test.Main.TestFiles.Concentration.Concentration (String description, int identifier, double value)`

6.8.2 Member Function Documentation

6.8.2.1 `int test.Main.TestFiles.Concentration.get_identifier ()`

6.8.2.2 `double test.Main.TestFiles.Concentration.get_value ()`

6.8.2.3 `String test.Main.TestFiles.Concentration.toString ()`

6.8.3 Member Data Documentation

6.8.3.1 `test.Main.TestFiles.Concentration.HIGH = ("High, [std deviation]", 0, 0)`

6.8.3.2 `test.Main.TestFiles.Concentration.LOW`

Initial value:

```
=("Low, [std deviation]", 2,
 0)
```

6.8.3.3 `test.Main.TestFiles.Concentration.MEDIUM = ("Medium, [std deviation]", 1, 0)`

The documentation for this enum was generated from the following file:

- [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/Main.java](#)

6.9 test.Main.TestFiles.Distribution Enum Reference

Public Member Functions

- [Distribution](#) (String *description*, int *identifier*, double *value*)
- String [toString](#) ()
- int [get_identifier](#) ()
- double [get_value](#) ()

Public Attributes

- [NORMAL](#) = ("Normal, centered about pressure median", 0, 0)
- [ABNORMAL](#)
- [RANDOM](#) = ("Random, completly and utterly random", 2, 0)

6.9.1 Constructor & Destructor Documentation

6.9.1.1 `test.Main.TestFiles.Distribution.Distribution (String description, int identifier, double value)`

6.9.2 Member Function Documentation

6.9.2.1 `int test.Main.TestFiles.Distribution.get_identifier ()`

6.9.2.2 `double test.Main.TestFiles.Distribution.get_value ()`

6.9.2.3 `String test.Main.TestFiles.Distribution.toString ()`

6.9.3 Member Data Documentation

6.9.3.1 `test.Main.TestFiles.Distribution.ABNORMAL`

Initial value:

```
= (
    "Abnormal, centered about pressure median, but inverted", 1,
    0)
```

6.9.3.2 `test.Main.TestFiles.Distribution.NORMAL = ("Normal, centered about pressure median", 0, 0)`

6.9.3.3 `test.Main.TestFiles.Distribution.RANDOM = ("Random, completely and utterly random", 2, 0)`

The documentation for this enum was generated from the following file:

- `/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/Main.java`

6.10 components.Distribution Class Reference

Used to compute and store max, min, std_deviation, and average for a list of [Touch](#).

Public Member Functions

- [Distribution](#) (List< [Touch](#) > touches)
- [Distribution](#) (List< [Touch](#) > touches, int keycode)
this constructor allows a keycode to be associated with the distribution
- [Distribution](#) ([Distribution](#) d)
copy constructor. This exists because computations are done in the constructor. Copying in this way avoids recomputation.
- void [update](#) (List< [Touch](#) > touches)
updates the distribution using a list of touches. This update has nothing to do with the old values in the distribution. It is synonymous to creating a new [Distribution](#) object with this list of touches.
- double [get_min](#) ()
- double [get_max](#) ()
- double [get_average](#) ()
- double [get_standard_deviation](#) ()
- int [get_keycode](#) ()
returns the keycode associated with this distribution. If the distribution does not have an associated keycode, this method will return -1.
- boolean [equals](#) (Object o)
determine if this distribution is exactly equal to another distribution

6.10.1 Detailed Description

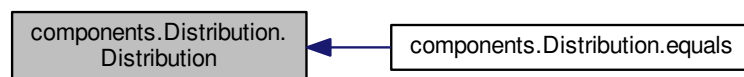
Used to compute and store max, min, std_deviation, and average for a list of [Touch](#).

In addition to computing these metrics, [Distribution](#) allows a keycode to be associated with the list of touches.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 `components.Distribution.Distribution (List< Touch > touches)`

Here is the caller graph for this function:



6.10.2.2 `components.Distribution.Distribution (List< Touch > touches, int keycode)`

this constructor allows a keycode to be associated with the distribution

6.10.2.3 `components.Distribution.Distribution (Distribution d)`

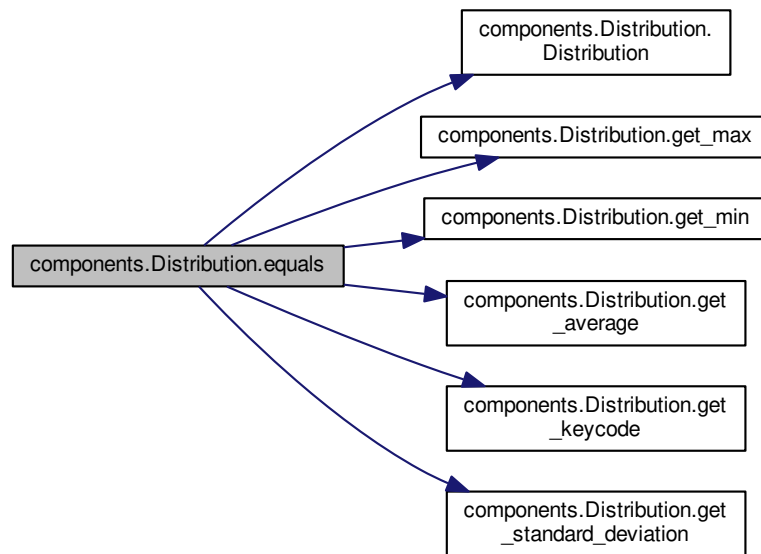
copy constructor. This exists because computations are done in the constructor. Copying in this way avoids recomputation.

6.10.3 Member Function Documentation

6.10.3.1 `boolean components.Distribution.equals (Object o)`

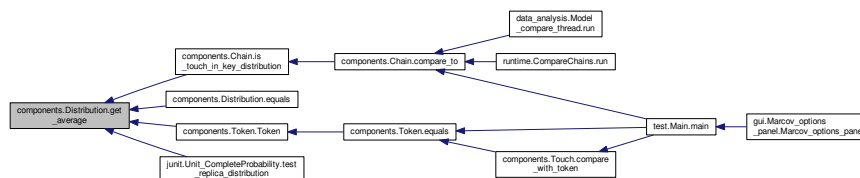
determine if this distribution is exactly equal to another distribution

Here is the call graph for this function:



6.10.3.2 double components.Distribution.get_average ()

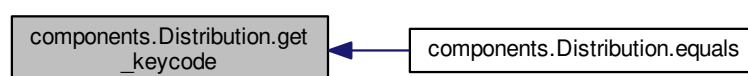
Here is the caller graph for this function:



6.10.3.3 int components.Distribution.get_keycode ()

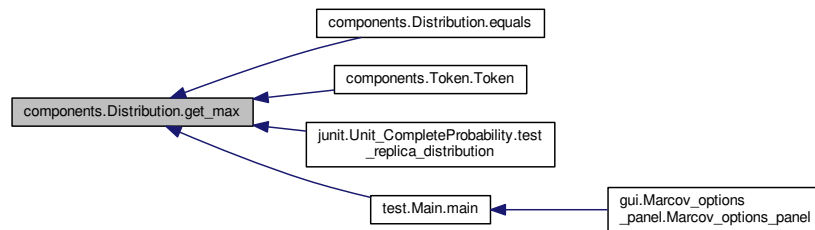
returns the keycode associated with this distribution. If the distribution does not have an associated keycode, this method will return -1.

Here is the caller graph for this function:



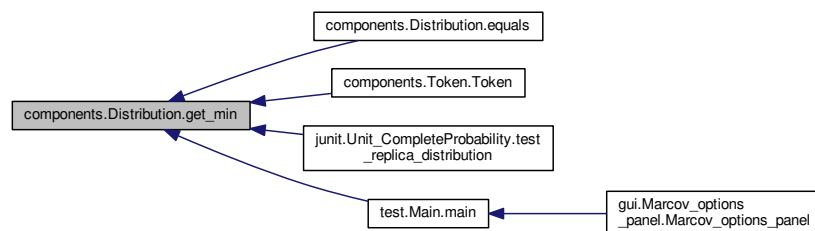
6.10.3.4 double components.Distribution.get_max ()

Here is the caller graph for this function:



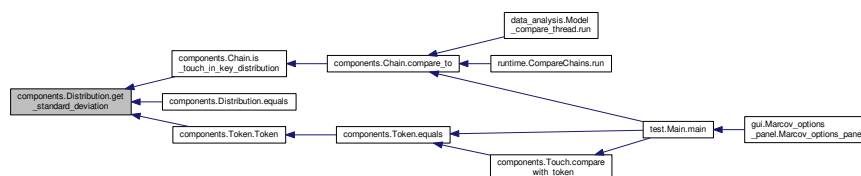
6.10.3.5 double components.Distribution.get_min ()

Here is the caller graph for this function:



6.10.3.6 double components.Distribution.get_standard_deviation ()

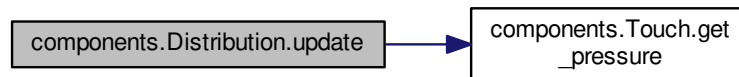
Here is the caller graph for this function:



6.10.3.7 void components.Distribution.update (List< Touch > touches)

updates the distribution using a list of touches. This update has nothing to do with the old values in the distribution. It is synonomous to creating a new [Distribution](#) object with this list of touches.

Here is the call graph for this function:



The documentation for this class was generated from the following file:

- [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/Distribution.java](#)

6.11 test.Main Class Reference

This class is used to test that the model is being built correctly.

Static Public Member Functions

- static void [main](#) (String args[])

6.11.1 Detailed Description

This class is used to test that the model is being built correctly.

Also tested is the model compairason. and various classes used in model creating. The idea is to print out the tests which fail. This class should have to do no actual work if the program is designed well.

NOTE: A few tests fail. This is not an issue as This is expected. The reason for this is that this test code is somewhat out of date.

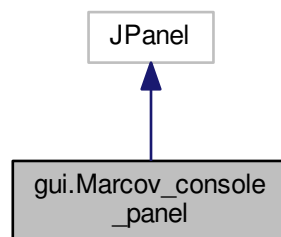
The documentation for this class was generated from the following file:

- [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/Main.java](#)

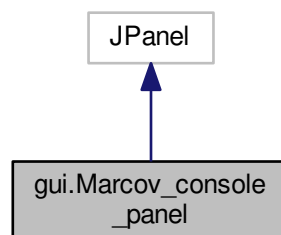
6.12 gui.Marcov_console_panel Class Reference

Displays messages printed to stdout.

Inheritance diagram for gui.Marcov_console_panel:



Collaboration diagram for gui.Marcov_console_panel:



Public Member Functions

- [Marcov_console_panel\(\)](#)

6.12.1 Detailed Description

Displays messages printed to stdout.

6.12.2 Constructor & Destructor Documentation

6.12.2.1 `gui.Marcov_console_panel.Marcov_console_panel ()`

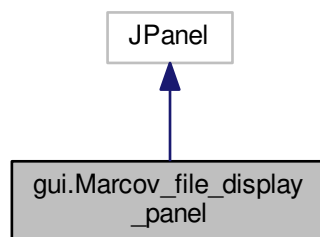
The documentation for this class was generated from the following file:

- `/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/Marcov_console_panel.java`

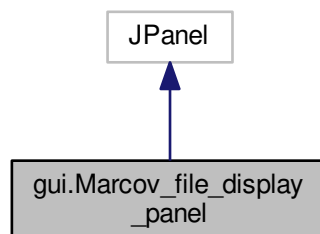
6.13 `gui.Marcov_file_display_panel` Class Reference

Displays files relevant to test code.

Inheritance diagram for `gui.Marcov_file_display_panel`:



Collaboration diagram for `gui.Marcov_file_display_panel`:



Public Member Functions

- `Marcov_file_display_panel ()`

6.13.1 Detailed Description

Displays files relevant to test code.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 gui.Marcov_file_display_panel.Marcov_file_display_panel ()

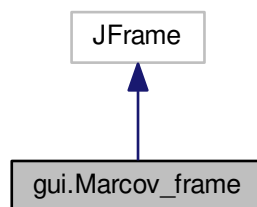
The documentation for this class was generated from the following file:

- [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/Marcov_file_display_panel.java](#)

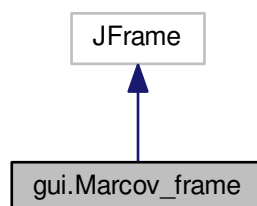
6.14 gui.Marcov_frame Class Reference

Display frame to contain buttons for running test code and panels to view results.

Inheritance diagram for gui.Marcov_frame:



Collaboration diagram for gui.Marcov_frame:



Public Member Functions

- [Marcov_frame](#) ()
- void [close](#) ()

6.14.1 Detailed Description

Display frame to contain buttons for running test code and panels to view results.

6.14.2 Constructor & Destructor Documentation

6.14.2.1 `gui.Marcov_frame.Marcov_frame ()`

6.14.3 Member Function Documentation

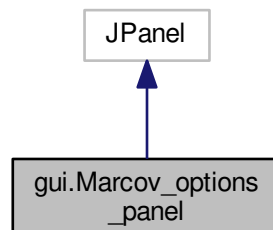
6.14.3.1 `void gui.Marcov_frame.close ()`

The documentation for this class was generated from the following file:

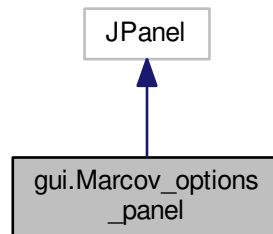
- `/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/Marcov_frame.java`

6.15 `gui.Marcov_options_panel` Class Reference

Inheritance diagram for `gui.Marcov_options_panel`:



Collaboration diagram for gui.Marcov_options_panel:



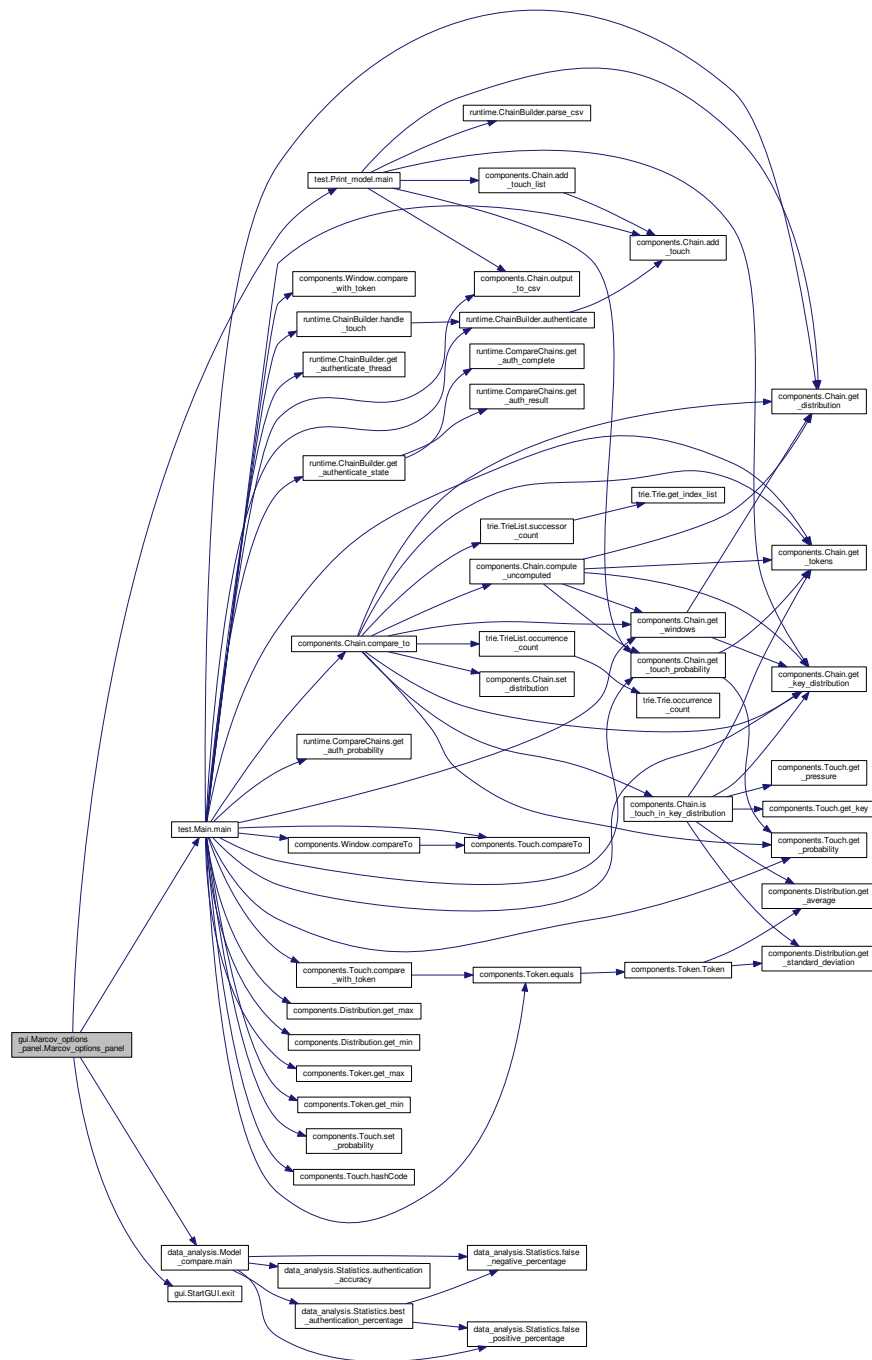
Public Member Functions

- [Marcov_options_panel\(\)](#)

6.15.1 Constructor & Destructor Documentation

6.15.1.1 gui.Marcov_options_panel.Marcov_options_panel ()

Here is the call graph for this function:



The documentation for this class was generated from the following file:

- /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/Marcov_options_panel.java

6.16 data_analysis.Model_compare Class Reference

Analysis class used to compare and analyze data gathered from users.

Static Public Member Functions

- static void [main](#) (String[] args)

6.16.1 Detailed Description

Analysis class used to compare and analyze data gathered from users.

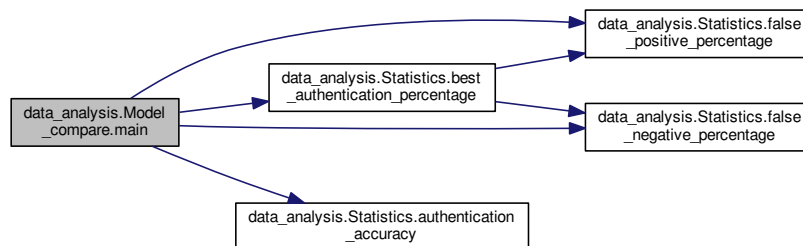
The purpose of this class is to test out the model compare process on data that has been collected The data to used will be contained in the data_sets folder input: data_sets folder output: model_compare_output.txt

6.16.2 Member Function Documentation

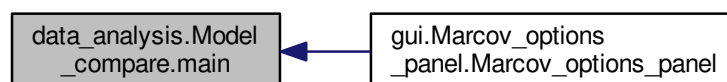
6.16.2.1 static void data_analysis.Model_compare.main (String[] args) [static]

create a number of tests with different parameters

Here is the call graph for this function:



Here is the caller graph for this function:



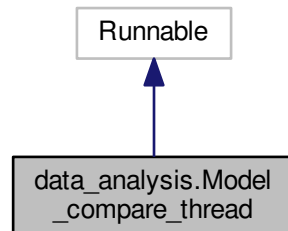
The documentation for this class was generated from the following file:

- `/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/data_analysis/Model_compare.java`

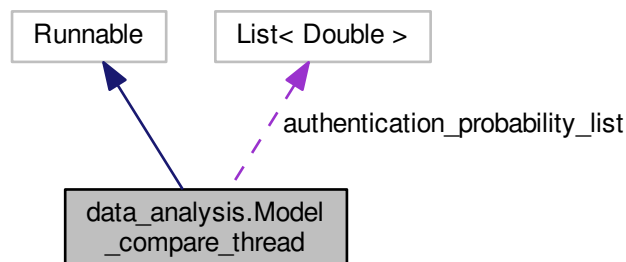
6.17 data_analysis.Model_compare_thread Class Reference

Compare Markov Chains on their own thread.

Inheritance diagram for data_analysis.Model_compare_thread:



Collaboration diagram for data_analysis.Model_compare_thread:



Public Member Functions

- [Model_compare_thread](#) (String base_data_path, String auth_data_path, int base_model_size, int auth_model_size, int window_size, int token_size, int threshold)
constructor, allowing user to set different propperties of the model compairason for testing
- void [run](#) ()
- String [get_base_data_path](#) ()
- String [get_auth_data_path](#) ()
- int [get_window_size](#) ()
- int [get_token_size](#) ()
- int [get_threshold](#) ()
- int [get_base_model_size](#) ()
- int [get_auth_model_size](#) ()
- List< Double > [get_auth_probability_list](#) ()

Public Attributes

- double [max_authentication_probability](#)
- double [min_authentication_probability](#)
- double [average_authentication_probability](#)

6.17.1 Detailed Description

Compare Markov Chains on their own thread.

This class was used to help speed up testing primarily. this thread allows the preforming of a test compairason. when the compairason is finished, an instance variable will be set indicating different results.

6.17.2 Constructor & Destructor Documentation

6.17.2.1 `data_analysis.Model_compare_thread.Model_compare_thread (String base_data_path, String auth_data_path, int base_model_size, int auth_model_size, int window_size, int token_size, int threshold)`

constructor, allowing user to set different propperties of the model compairason for testing

6.17.3 Member Function Documentation

6.17.3.1 `String data_analysis.Model_compare_thread.get_auth_data_path ()`

6.17.3.2 `int data_analysis.Model_compare_thread.get_auth_model_size ()`

6.17.3.3 `List<Double> data_analysis.Model_compare_thread.get_auth_probability_list ()`

6.17.3.4 `String data_analysis.Model_compare_thread.get_base_data_path ()`

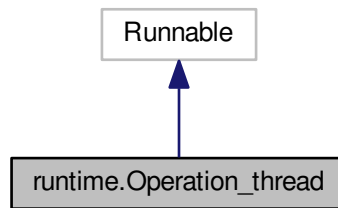
6.17.3.5 `int data_analysis.Model_compare_thread.get_base_model_size ()`

6.17.3.6 `int data_analysis.Model_compare_thread.get_threshold ()`

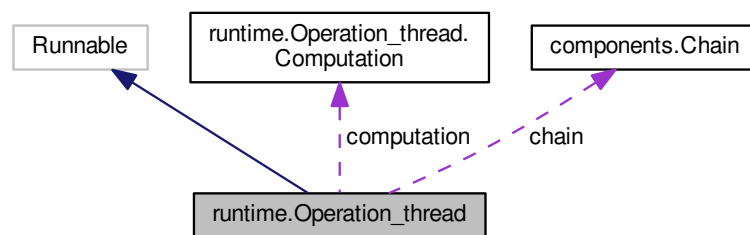
6.17.3.7 `int data_analysis.Model_compare_thread.get_token_size ()`

6.17.3.8 `int data_analysis.Model_compare_thread.get_window_size ()`

Inheritance diagram for runtime.Operation_thread:



Collaboration diagram for runtime.Operation_thread:



Classes

- enum [Computation](#)

Public Member Functions

- [Operation_thread](#) ([Chain](#) chain, [Computation](#) computation)
- void [run](#) ()

6.18.1 Detailed Description

UNUSED.

The Intent of this class was to run specific computations on a differant thread. It is unused in the current implemen-
tation because there is never a need to run one computation independent from the others.

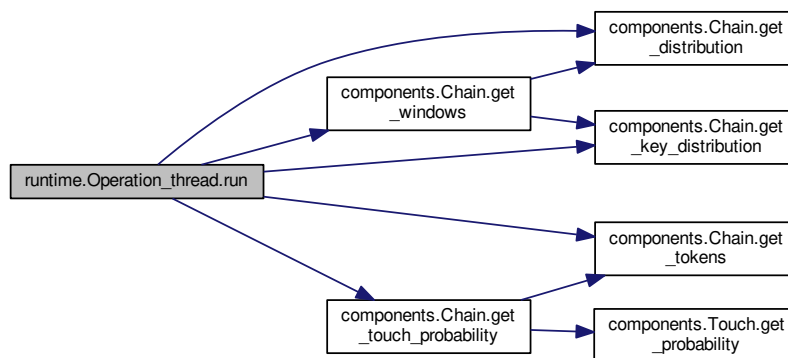
6.18.2 Constructor & Destructor Documentation

6.18.2.1 `runtime.Operation_thread.Operation_thread (Chain chain, Computation computation)`

6.18.3 Member Function Documentation

6.18.3.1 `void runtime.Operation_thread.run ()`

Here is the call graph for this function:



The documentation for this class was generated from the following file:

- `/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/runtime/Operation_thread.java`

6.19 test.Main.TestFiles.PressureAmount Enum Reference

Public Member Functions

- `PressureAmount` (String description, int identifier, double value)
- `String toString ()`
- `int get_identifier ()`
- `double get_value ()`

Public Attributes

- `HIGH` = ("High pressure, 0.75", 0, .75)
- `MEDIUM` = ("Medium Pressure, 0.5", 1, .5)
- `LOW`

6.19.1 Constructor & Destructor Documentation

6.19.1.1 `test.Main.TestFiles.PressureAmount.PressureAmount (String description, int identifier, double value)`

6.19.2 Member Function Documentation

6.19.2.1 `int test.Main.TestFiles.PressureAmount.get_identifier ()`

6.19.2.2 `double test.Main.TestFiles.PressureAmount.get_value ()`

6.19.2.3 `String test.Main.TestFiles.PressureAmount.toString ()`

6.19.3 Member Data Documentation

6.19.3.1 `test.Main.TestFiles.PressureAmount.HIGH =("High pressure, 0.75", 0, .75)`

6.19.3.2 `test.Main.TestFiles.PressureAmount.LOW`

Initial value:

```
=("Low Pressure, 0.25", 2,
  .25)
```

6.19.3.3 `test.Main.TestFiles.PressureAmount.MEDIUM =("Medium Pressure, 0.5", 1, .5)`

The documentation for this enum was generated from the following file:

- `/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/Main.java`

6.20 test.Print_model Class Reference

This class will print out the model constructed form the designated file.

Static Public Member Functions

- static void `main` (String[] args)

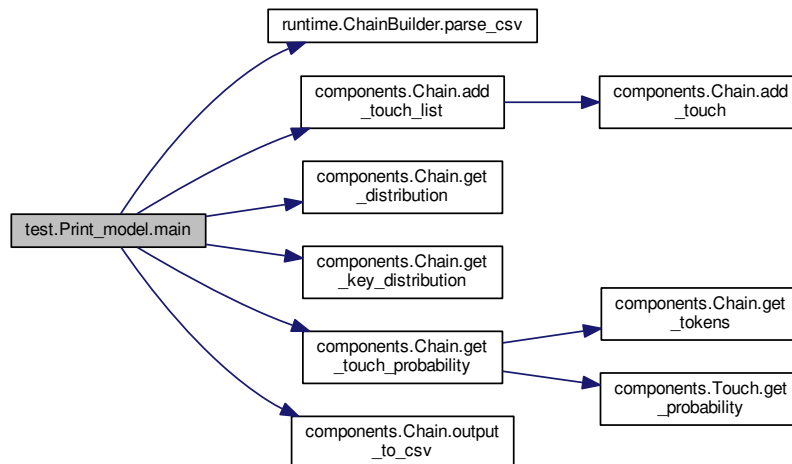
6.20.1 Detailed Description

This class will print out the model constructed form the designated file.

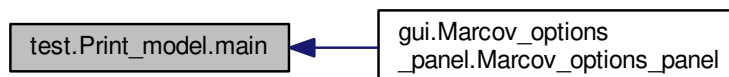
6.20.2 Member Function Documentation

6.20.2.1 static void test.Print_model.main (String[] args) [static]

Here is the call graph for this function:



Here is the caller graph for this function:



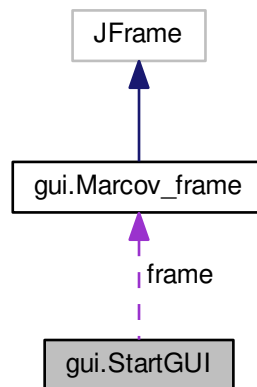
The documentation for this class was generated from the following file:

- `/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/Print_model.java`

6.21 gui.StartGUI Class Reference

GUI useful for testing.

Collaboration diagram for gui.StartGUI:



Static Public Member Functions

- static void `main` (String[] args)
- static void `exit` ()
causes the frame to close

6.21.1 Detailed Description

GUI useful for testing.

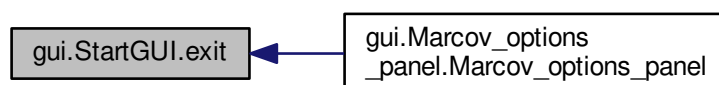
Contains functionality to run test code and view results.

6.21.2 Member Function Documentation

6.21.2.1 static void gui.StartGUI.exit () [static]

causes the frame to close

Here is the caller graph for this function:



6.21.2.2 static void gui.StartGUI.main (String[] args) [static]

The documentation for this class was generated from the following file:

- /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/[StartGUI.java](#)

6.22 runtime.ChainBuilder.State Enum Reference

Public Attributes

- [IN_PROGRESS](#)
- [SUCCESS](#)

6.22.1 Member Data Documentation

6.22.1.1 runtime.ChainBuilder.State.IN_PROGRESS

6.22.1.2 runtime.ChainBuilder.State.SUCCESS

The documentation for this enum was generated from the following file:

- /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/runtime/[ChainBuilder.java](#)

6.23 data_analysis.Statistics Class Reference

Generates statistics on results generated by model_compare.java.

Static Public Member Functions

- static void [main](#) (String args[])
- static double [false_positive_percentage](#) (double authentication_percentage, List< Double > should_authenticate_percentages, List< Double > should_not_authenticate_percentages)
- static double [false_negative_percentage](#) (double authentication_percentage, List< Double > should_authenticate_percentages, List< Double > should_not_authenticate_percentages)
- static double [best_authentication_percentage](#) (List< Double > should_authenticate_percentages, List< Double > should_not_authenticate_percentages)
- static double [minimize_false_positive_authentication_percentage](#) (List< Double > should_authenticate_percentages, List< Double > should_not_authenticate_percentages)
- static double [equal_false_positive_negative_authentication_percentage](#) (List< Double > should_authenticate_percentages, List< Double > should_not_authenticate_percentages)
- static double [authentication_accuracy](#) (double authentication_percentage, List< Double > should_authenticate_percentages, List< Double > should_not_authenticate_percentages)

6.23.1 Detailed Description

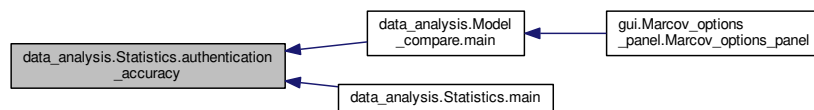
Generates statistics on results generated by model_compare.java.

NOTE: there are bugs in the maximize, minimize functions for authentication accuracy. I did min/max authentication accuracy analysis in another program after the fact. The results on authentication accuracy, false positive, and false negative may be trusted though.

6.23.2 Member Function Documentation

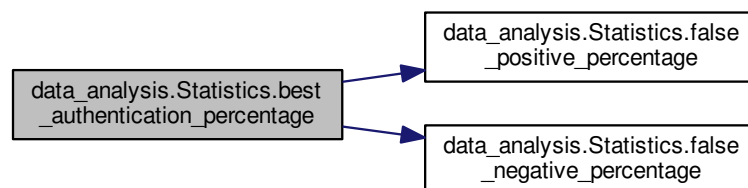
6.23.2.1 static double data_analysis.Statistics.authentication_accuracy (double authentication_percentage, List< Double > should_authenticate_percentages, List< Double > should_not_authenticate_percentages) [static]

Here is the caller graph for this function:

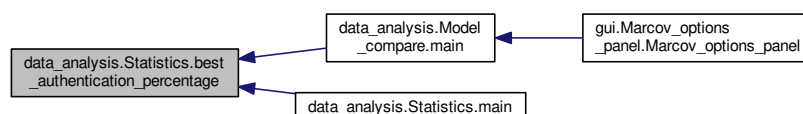


6.23.2.2 static double data_analysis.Statistics.best_authentication_percentage (List< Double > should_authenticate_percentages, List< Double > should_not_authenticate_percentages) [static]

Here is the call graph for this function:

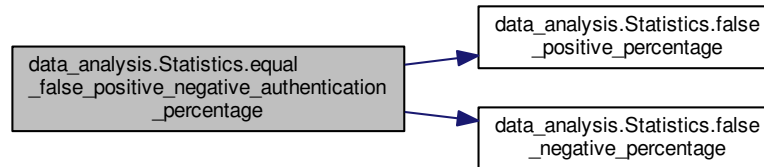


Here is the caller graph for this function:



6.23.2.3 `static double data_analysis.Statistics.equal_false_positive_negative_authentication_percentage (List< Double > should_authenticate_percentages, List< Double > should_not_authenticate_percentages) [static]`

Here is the call graph for this function:

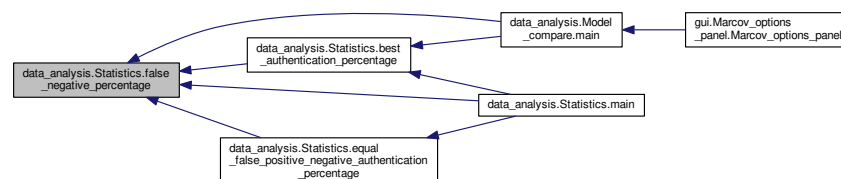


Here is the caller graph for this function:



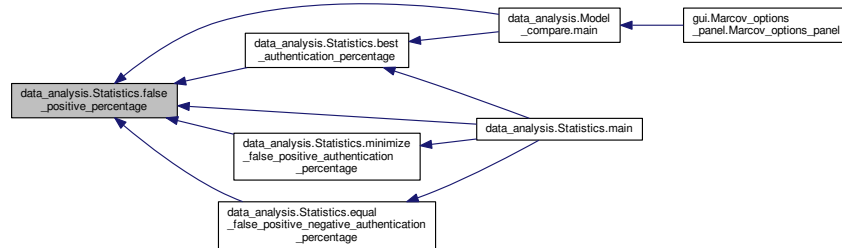
6.23.2.4 `static double data_analysis.Statistics.false_negative_percentage (double authentication_percentage, List< Double > should_authenticate_percentages, List< Double > should_not_authenticate_percentages) [static]`

Here is the caller graph for this function:



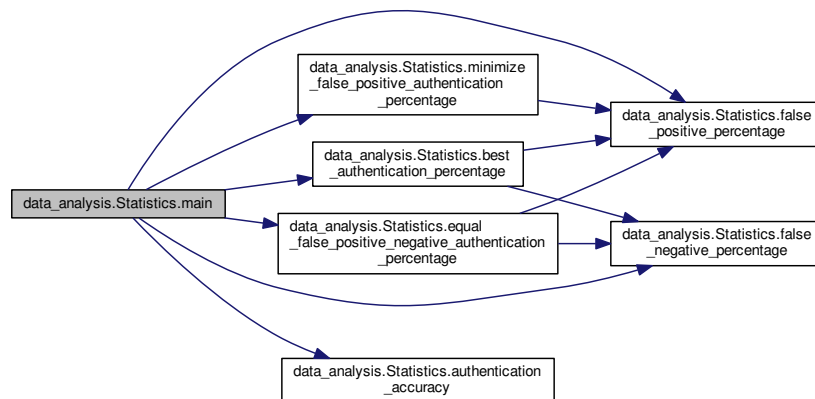
6.23.2.5 static double data_analysis.Statistics.false_positive_percentage (double *authentication_percentage*, List< Double > *should_authenticate_percentages*, List< Double > *should_not_authenticate_percentages*) [static]

Here is the caller graph for this function:



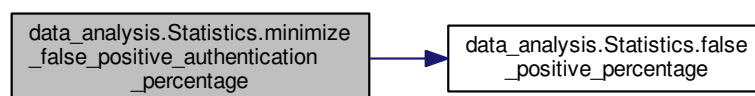
6.23.2.6 static void data_analysis.Statistics.main (String *args*[]) [static]

Here is the call graph for this function:

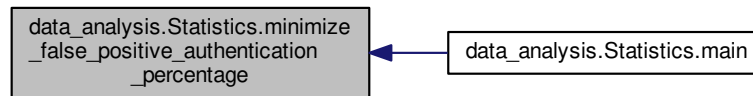


6.23.2.7 static double data_analysis.Statistics.minimize_false_positive_authentication_percentage (List< Double > *should_authenticate_percentages*, List< Double > *should_not_authenticate_percentages*) [static]

Here is the call graph for this function:



Here is the caller graph for this function:



The documentation for this class was generated from the following file:

- [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/data_analysis/Statistics.java](#)

6.24 components.Token Class Reference

This class represents a token within the model.

Classes

- enum [Type](#)
specify the type of token we want to build

Public Member Functions

- [Token](#) ([Distribution](#) distribution, int total_tokens, int token_index, double standard_deviations, [Type](#) type)
allow tokens to be created making each touch mu for its keycode, or in a linear fashion over the distribution for the keycode
- [Token](#) ([Distribution](#) distribution, int total_tokens, int token_index, [Type](#) type)
allow for creation of tokens over a distribution
- [Token](#) (double range_min, double range_max, int total_tokens, int token_index, [Type](#) type)
Implemented in the constructor is the clustering algorithm.
- boolean [contains](#) ([Touch](#) touch)
determines if a touch is within this token based on its pressure value this will return true if a touches pressure equals max or min, so if max of one token is min of another token, both will return true
- boolean [is_high_wildcard](#) ([Touch](#) touch)
determine if a touch is a high_wildcard
- boolean [is_low_wildcard](#) ([Touch](#) touch)
determine if a touch is a low wildcard
- void [increment_high_wildcards](#) ()
adds the the number of high wildcards
- void [increment_low_wildcards](#) ()
adds the the number of high wildcards
- int [get_total_wildcards](#) ()
returns the total number of wildcards
- int [get_acceptable_wildcards](#) (int total_items)

- returns the acceptable number of wildcards*
 - double `get_min` ()
- returns true if there are more than an acceptable number of wildcards*
 - double `get_max` ()
- return the maximum*
 - boolean `equals` (Object o_t)
- compares This token to another to determine if they are the same*

6.24.1 Detailed Description

This class represents a token within the model.

Essentially this is a range of values. A touch is defined to be within a token if the pressure value of the touch falls within this range. This class is designed to abstract away the clustering algorithm. This makes the rest of the code far simpler to think about. Something to look at in the future may be a clustering algorithm that is not equally distributed.

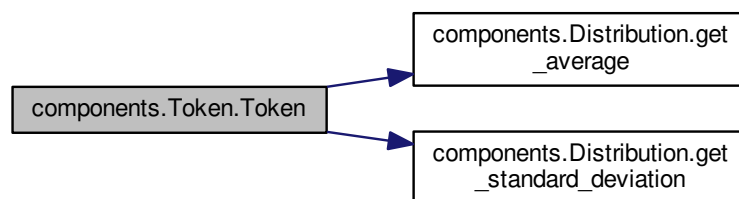
6.24.2 Constructor & Destructor Documentation

6.24.2.1 `components.Token.Token (Distribution distribution, int total_tokens, int token_index, double standard_deviations, Type type)`

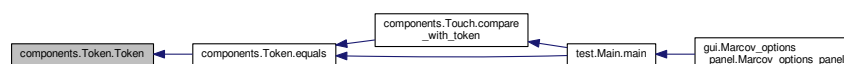
allow tokens to be created making each touch mu for its keycode, or in a linear fashion over the distribution for the keycode

allow for creation of tokens with-in some number of standard deviations of a distribution

Here is the call graph for this function:



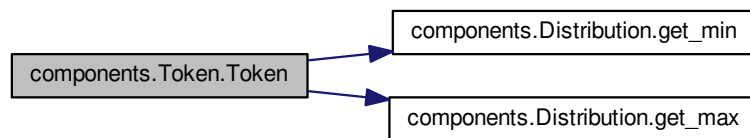
Here is the caller graph for this function:



6.24.2.2 components.Token.Token (**Distribution** *distribution*, int *total_tokens*, int *token_index*, **Type** *type*)

allow for creation of tokens over a distribution

Here is the call graph for this function:



6.24.2.3 components.Token.Token (double *range_min*, double *range_max*, int *total_tokens*, int *token_index*, **Type** *type*)

Implemented in the constructor is the clustering algorithm.

This determines how to split up the range into a number of tokens.

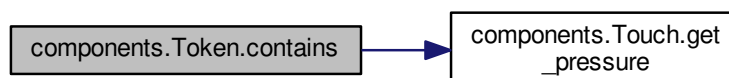
range_min, minimum of the token range *range_max*, maximum of the token range *total_tokens* total number of tokens to split range into *token_index*, integer between 0 and *total_tokens*-1 indicating into which range this token falls

6.24.3 Member Function Documentation

6.24.3.1 boolean components.Token.contains (**Touch** *touch*)

determines if a touch is within this token based on its pressure value this will return true if a touches pressure equals max or min, so if max of one token is min of another token, both will return true

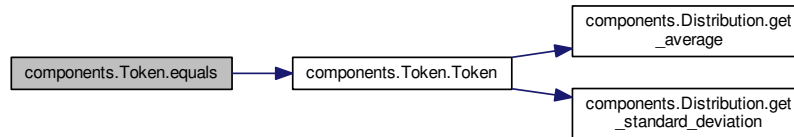
Here is the call graph for this function:



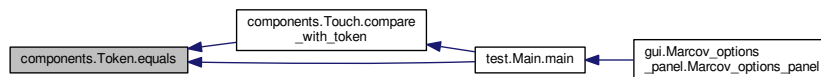
6.24.3.2 boolean components.Token.equals (Object o_t)

compares This token to another to determine if they are the same

Here is the call graph for this function:



Here is the caller graph for this function:



6.24.3.3 int components.Token.get_acceptable_wildcards (int total_items)

returns the acceptable number of wildcards

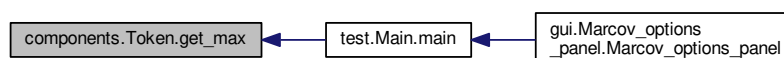
Parameters

<code>total_items</code>	is the total number of items in the distribution
--------------------------	--

6.24.3.4 double components.Token.get_max ()

return the maximum

Here is the caller graph for this function:

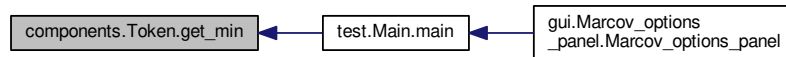


6.24.3.5 double components.Token.get_min ()

returns true if there are more than an acceptable number of wildcards

return minimum

Here is the caller graph for this function:



6.24.3.6 int components.Token.get_total_wildcards ()

returns the total number of wildcards

6.24.3.7 void components.Token.increment_high_wildcards ()

adds the the number of high wildcards

6.24.3.8 void components.Token.increment_low_wildcards ()

adds the the number of high wildcards

6.24.3.9 boolean components.Token.is_high_wildcard (Touch touch)

determine if a touch is a high_wildcard

Here is the call graph for this function:



6.24.3.10 boolean components.Token.is_low_wildcard (Touch touch)

determine if a touch is a low wildcard

Here is the call graph for this function:



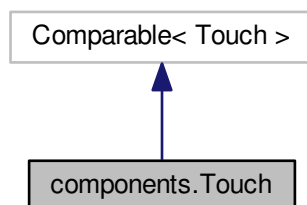
The documentation for this class was generated from the following file:

- `/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/Token.java`

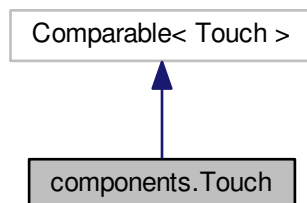
6.25 components.Touch Class Reference

This class represents a touch event.

Inheritance diagram for components.Touch:



Collaboration diagram for components.Touch:



Public Member Functions

- [Touch](#) (int keycode, double pressure, long timestamp)
- [Touch](#) ([Touch](#) t)
copy constructor
- void [set_probability](#) ([Window](#) preceeding_window, double p)
sets the probability that this touch succeeds a given sequence. Reccord the sequence and the probability
- double [get_probability](#) ([Window](#) preceeding_window)
returns the probability of the touch occurring after a given window w. If the window does not exist return (TODO) currently returning 0
- double [get_pressure](#) ()
- int [get_key](#) ()
- long [get_timestamp](#) ()
- boolean [compare_with_token](#) (List< [Token](#) > tokens, [Touch](#) other_touch)
compares the touches with the given token list.
- int [hashCode](#) ()
implement hash function for the touch class
- int [compareTo](#) ([Touch](#) other_touch)
compare touches to one another. return negative if this touch is less than other_touch
- String [toString](#) ()

6.25.1 Detailed Description

This class represents a touch event.

[Touch](#) evens have an associated key, pressure, and time from the raw data. From the Markov [Chain](#) we derive the probability and predecessor_window attributes. The probability is a value between 0 and 1 representing the percent change this token follows the predecessor window.

6.25.2 Constructor & Destructor Documentation

6.25.2.1 components.Touch.Touch (int *keycode*, double *pressure*, long *timestamp*)

6.25.2.2 components.Touch.Touch ([Touch](#) t)

copy constructor

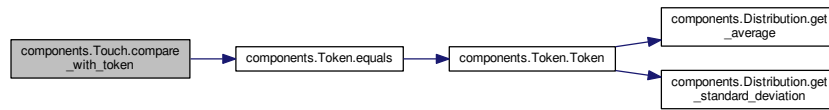
6.25.3 Member Function Documentation

6.25.3.1 boolean components.Touch.compare_with_token (List< [Token](#) > *tokens*, [Touch](#) *other_touch*)

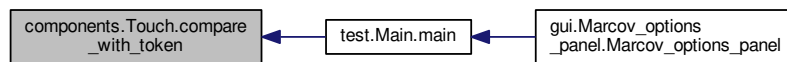
compares the touches with the given token list.

this function will return true if the touches are contained within the smae token

Here is the call graph for this function:



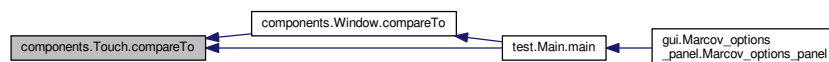
Here is the caller graph for this function:



6.25.3.2 int components.Touch.compareTo (Touch *other_touch*)

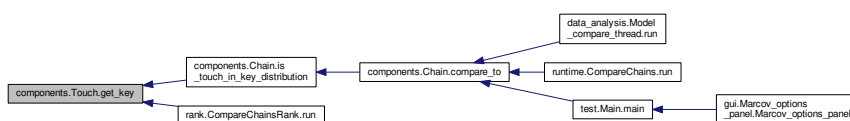
compare touches to one another. return negative if this touch is less than other_touch

Here is the caller graph for this function:



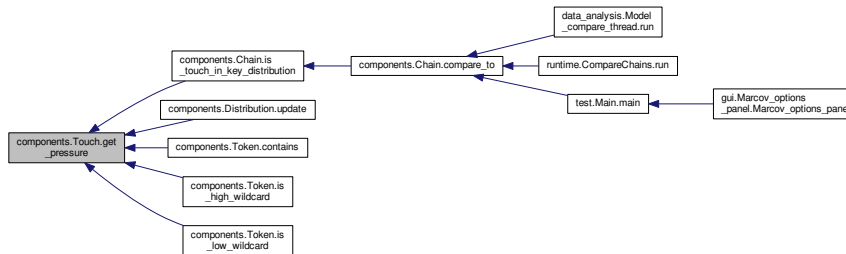
6.25.3.3 int components.Touch.get_key ()

Here is the caller graph for this function:



6.25.3.4 double components.Touch.get_pressure ()

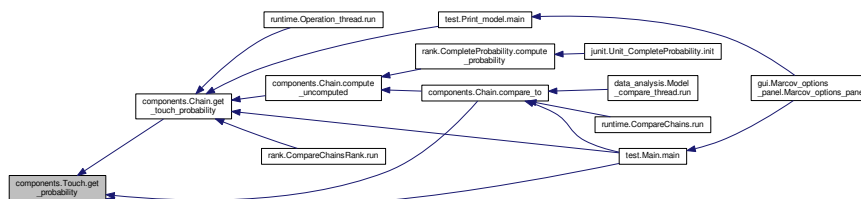
Here is the caller graph for this function:



6.25.3.5 double components.Touch.get_probability (Window preceeding_window)

returns the probability of the touch occurring after a given window w. If the window does not exist return (TODO) currently returning 0

Here is the caller graph for this function:

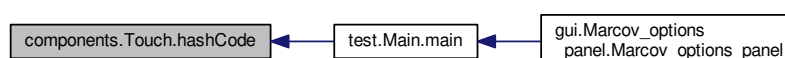


6.25.3.6 long components.Touch.get_timestamp ()

6.25.3.7 int components.Touch.hashCode ()

implement hash function for the touch class

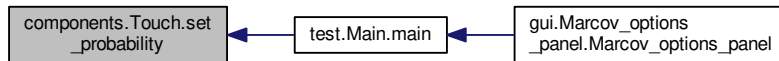
Here is the caller graph for this function:



6.25.3.8 void components.Touch.set_probability (Window preceeding_window, double p)

sets the probability that this touch succeeds a given sequence. Reccord the sequence and the probability

Here is the caller graph for this function:



6.25.3.9 String components.Touch.toString ()

The documentation for this class was generated from the following file:

- /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/[Touch.java](#)

6.26 trie.Trie Class Reference

Implementation of Prefix Tree.

Classes

- class **TrieNode**

Public Member Functions

- [Trie](#) ()
sets up the tree so that everything will be added to trienode root?
- [Trie](#) ([Trie](#) t)
creates a copy trie
- void [clear](#) ()
removes all elements from the trie
- void [insertString](#) (String s, int index)
- int [occurrence_count](#) (String s)
retrieves the number of occurrences of a given string in the tree
- List< Integer > [get_index_list](#) (String s)
returns a list of indexes containing the given window
- void [printSorted](#) (TrieNode node, String s)
prints the elements in a sorted order

6.26.1 Detailed Description

Implementation of Prefix Tree.

This benefits the efficiency of the program. This class is used primarily to figure out information about windows needed in the probability computation.

6.26.2 Constructor & Destructor Documentation

6.26.2.1 `trie.Trie.Trie ()`

sets up the tree so that everything will be added to trienode root?

6.26.2.2 `trie.Trie.Trie (Trie t)`

creates a copy trie

6.26.3 Member Function Documentation

6.26.3.1 `void trie.Trie.clear ()`

removes all elements from the trie

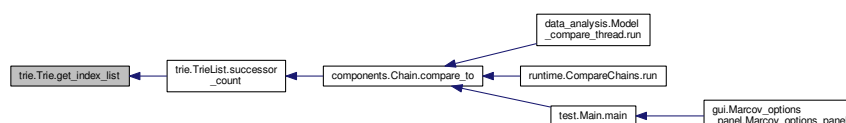
Here is the caller graph for this function:



6.26.3.2 `List<Integer> trie.Trie.get_index_list (String s)`

returns a list of indexes containing the given window

Here is the caller graph for this function:



6.26.3.3 void trie.Trie.insertString (String *s*, int *index*)

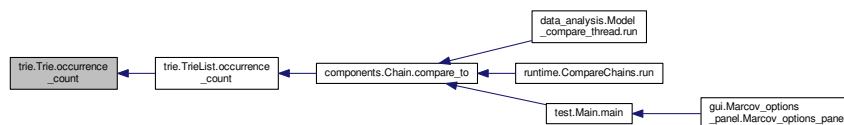
Here is the caller graph for this function:



6.26.3.4 int trie.Trie.occurrence_count (String *s*)

retrieves the number of occurrences of a given string in the tree

Here is the caller graph for this function:



6.26.3.5 void trie.Trie.printSorted (TrieNode *node*, String *s*)

prints the elements in a sorted order

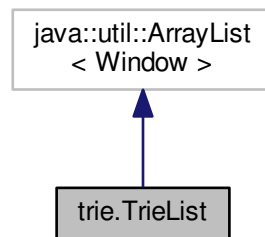
The documentation for this class was generated from the following file:

- `/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/trie/Trie.java`

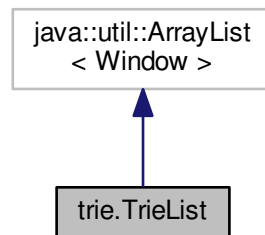
6.27 trie.TrieList Class Reference

Wrapper around [Trie](#) used to maintain an ordering among the stored elements.

Inheritance diagram for trie.TrieList:



Collaboration diagram for trie.TrieList:



Public Member Functions

- `TrieList ()`
- `TrieList (TrieList t)`
- `boolean add (Window arg0)`
- `void add (int arg0, Window arg1)`
- `boolean addAll (Collection<? extends Window > arg0)`
- `boolean addAll (int arg0, Collection<? extends Window > arg1)`
- `void clear ()`
- `boolean remove (Object arg0)`
- `Window remove (int arg0)`
- `boolean removeAll (Collection<?> arg0)`
- `boolean retainAll (Collection<?> arg0)`
- `Window set (int arg0, Window arg1)`
- `int successor_count (List< Touch > successor_list, Window window, Touch touch)`
counts the number of times a given touch comes after a given window. in the given window, succesors list
- `int occurrence_count (Window w)`
return the number of occurrences of w in window_list TODO I think this method needs to be faster.
- `void set_tokens (List< Token > tokens)`
sets the tokens that will be used when encoding the window

6.27.1 Detailed Description

Wrapper around [Trie](#) used to maintain an ordering among the stored elements.

This class uses some additional space to store elements in both an ArrayList and the prefix tree. NOTE: This was done for speed of implementation. It would be good if in the future only a prefix tree was used.

6.27.2 Constructor & Destructor Documentation

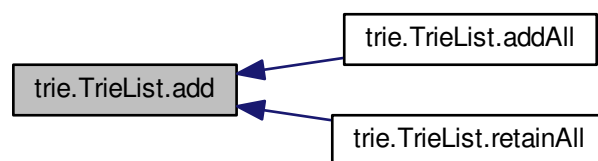
6.27.2.1 `trie.TrieList.TrieList ()`

6.27.2.2 `trie.TrieList.TrieList (TrieList t)`

6.27.3 Member Function Documentation

6.27.3.1 `boolean trie.TrieList.add (Window arg0)`

Here is the caller graph for this function:



6.27.3.2 `void trie.TrieList.add (int arg0, Window arg1)`

6.27.3.3 `boolean trie.TrieList.addAll (Collection<?extends Window > arg0)`

Here is the call graph for this function:



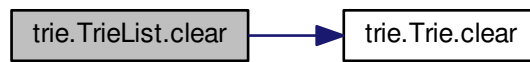
6.27.3.4 `boolean trie.TrieList.addAll (int arg0, Collection<?extends Window > arg1)`

Here is the call graph for this function:



6.27.3.5 `void trie.TrieList.clear ()`

Here is the call graph for this function:



Here is the caller graph for this function:

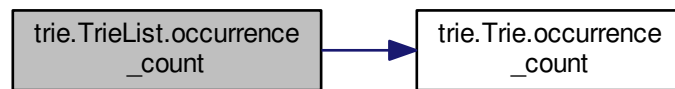


6.27.3.6 `int trie.TrieList.occurrence_count (Window w)`

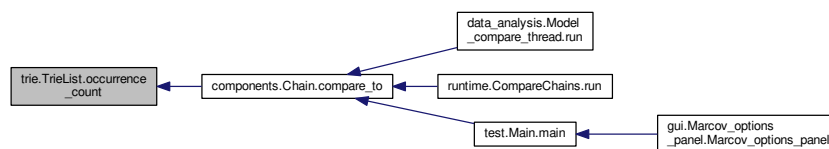
return the number of occurrences of w in window_list TODO I think this method needs to be faster.

Storing windows in a prefix tree would allow for this

Here is the call graph for this function:



Here is the caller graph for this function:



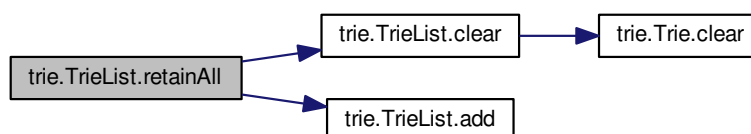
6.27.3.7 `boolean trie.TrieList.remove (Object arg0)`

6.27.3.8 `Window trie.TrieList.remove (int arg0)`

6.27.3.9 `boolean trie.TrieList.removeAll (Collection<?> arg0)`

6.27.3.10 `boolean trie.TrieList.retainAll (Collection<?> arg0)`

Here is the call graph for this function:

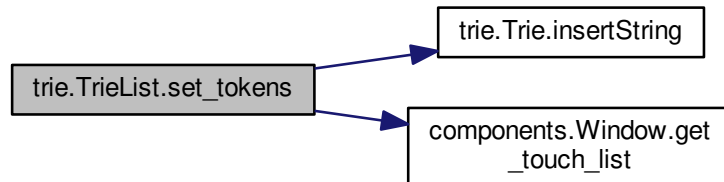


6.27.3.11 Window trie.TrieList.set (int *arg0*, Window *arg1*)

6.27.3.12 void trie.TrieList.set_tokens (List< Token > *tokens*)

sets the tokens that will be used when encoding the window

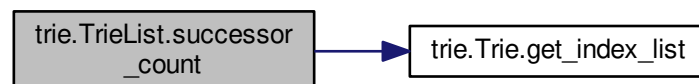
Here is the call graph for this function:



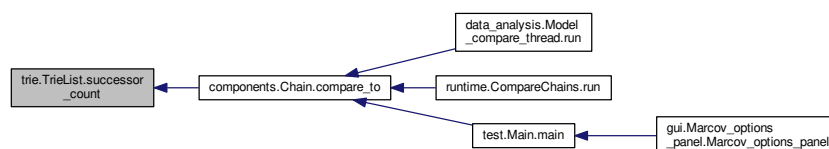
6.27.3.13 int trie.TrieList.successor_count (List< Touch > *successor_list*, Window *window*, Touch *touch*)

counts the number of times a given touch comes after a given window. in the given window, successors list

Here is the call graph for this function:



Here is the caller graph for this function:



The documentation for this class was generated from the following file:

- /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/trie/[TrieList.java](#)

6.28 components.Token.Type Enum Reference

specify the type of token we want to build

Public Attributes

- [linear](#)
- [keycode_mu](#)
- [combined](#)

6.28.1 Detailed Description

specify the type of token we want to build

6.28.2 Member Data Documentation

6.28.2.1 components.Token.Type.combined

6.28.2.2 components.Token.Type.keycode_mu

6.28.2.3 components.Token.Type.linear

The documentation for this enum was generated from the following file:

- [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/Token.java](#)

6.29 junit.Unit_CompareChainsRank Class Reference

goal is to test compare chains rank functionality

Public Member Functions

- void [init](#) ()
- void [test_authentication_probability](#) ()

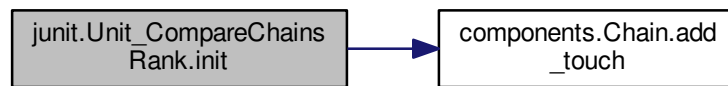
6.29.1 Detailed Description

goal is to test compare chains rank functionality

6.29.2 Member Function Documentation

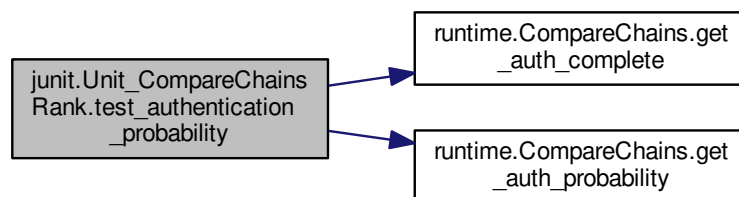
6.29.2.1 void junit.Unit_CompareChainsRank.init ()

Here is the call graph for this function:



6.29.2.2 void junit.Unit_CompareChainsRank.test_authentication_probability ()

Here is the call graph for this function:



The documentation for this class was generated from the following file:

- [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/junit/Unit_CompareChainsRank.java](#)

6.30 junit.Unit_CompleteProbability Class Reference

unit test demonstrating how to compute probability

Public Member Functions

- void [init](#) ()
- void [test_replica_distribution](#) ()

test different properties of replica chain to see if this works as expected.

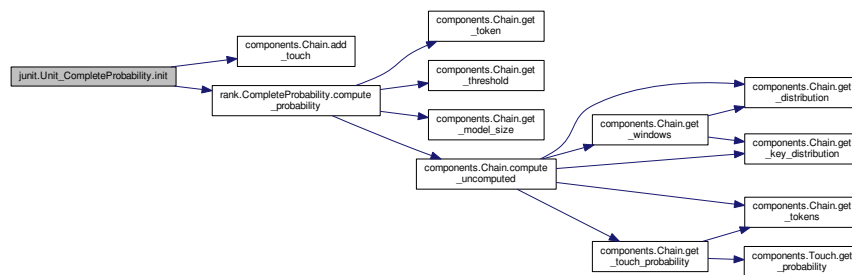
6.30.1 Detailed Description

unit test demonstrating how to compute probability

6.30.2 Member Function Documentation

6.30.2.1 void junit.Unit_CompleteProbability.init ()

Here is the call graph for this function:

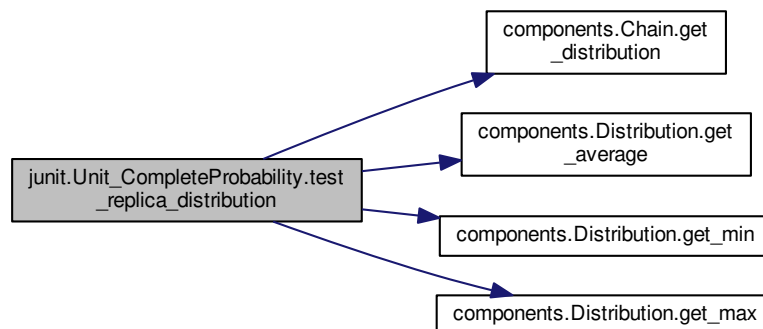


6.30.2.2 void junit.Unit_CompleteProbability.test_replica_distribution ()

test different properties of replica chain to see if this works as expected.

Replica chain should contain the probabilities for when window is equal to 1.

Here is the call graph for this function:



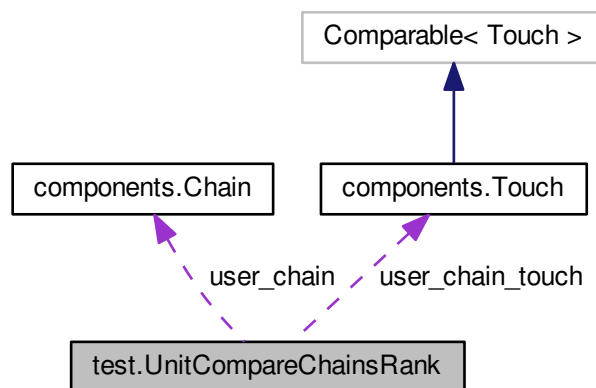
The documentation for this class was generated from the following file:

- /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/junit/[Unit_CompleteProbability.java](#)

6.31 test.UnitCompareChainsRank Class Reference

JUnit test for testing PageRank version of compairason Not that the PageRank implementation is not currently functional.

Collaboration diagram for test.UnitCompareChainsRank:



Public Member Functions

- void `init ()`
- void `test_chain_to_graph ()`
tests chain_to_graph method to make sure the chain is converted to a StateGraph correctly.
- void `test_touch_index ()`
make sure touch index returns the correct index in the list
- void `test_touch_window ()`
make sure the touch_window() returns the correct window in chain.
- void `test ()`
example test

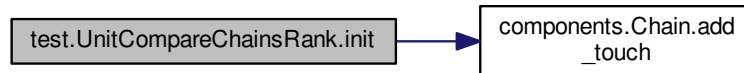
6.31.1 Detailed Description

JUnit test for testing PageRank version of compairason Not that the PageRank implementation is not currently functional.

6.31.2 Member Function Documentation

6.31.2.1 void test.UnitCompareChainsRank.init ()

Here is the call graph for this function:



6.31.2.2 void test.UnitCompareChainsRank.test ()

example test

6.31.2.3 void test.UnitCompareChainsRank.test_chain_to_graph ()

tests `chain_to_graph` method to make sure the chain is converted to a `StateGraph` correctly.

6.31.2.4 void test.UnitCompareChainsRank.test_touch_index ()

make sure touch index returns the correct index in the list

6.31.2.5 void test.UnitCompareChainsRank.test_touch_window ()

make sure the `touch_window()` returns the correct window in chain.

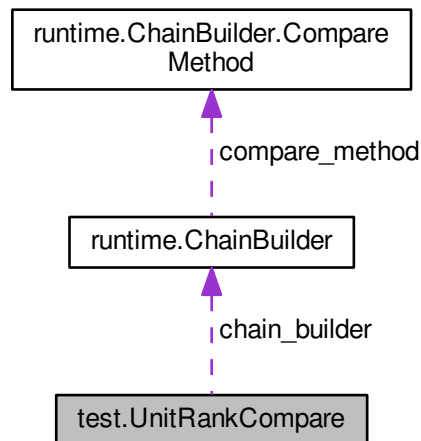
The documentation for this class was generated from the following file:

- `/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/UnitCompareChainsRank.java`

6.32 test.UnitRankCompare Class Reference

Test the compairason with ranks.

Collaboration diagram for test.UnitRankCompare:



Public Member Functions

- void `init` ()
- void `test_compare_correct` ()
checks that the probabilities are correct
- void `test_auth_probability` ()
check the test that the compare vectors are correct
- void `test` ()
example test

6.32.1 Detailed Description

Test the compairason with ranks.

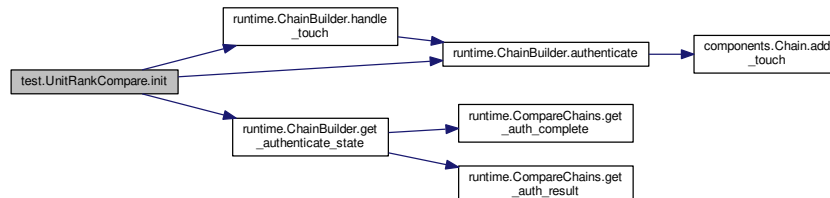
Author

element

6.32.2 Member Function Documentation

6.32.2.1 void test.UnitRankCompare.init ()

Here is the call graph for this function:



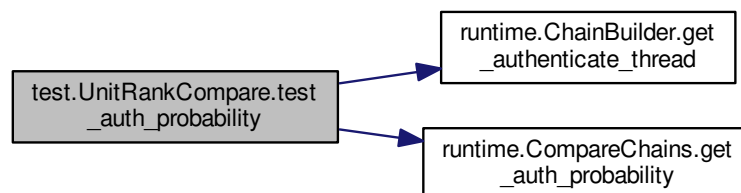
6.32.2.2 void test.UnitRankCompare.test ()

example test

6.32.2.3 void test.UnitRankCompare.test_auth_probability ()

check the test that the compare vectors are correct

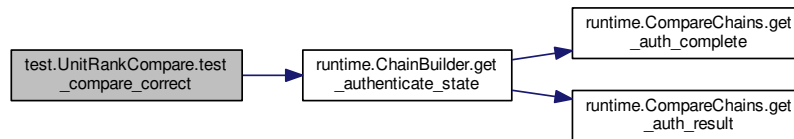
Here is the call graph for this function:



6.32.2.4 void test.UnitRankCompare.test_compare_correct ()

checks that the probabilities are correct

Here is the call graph for this function:



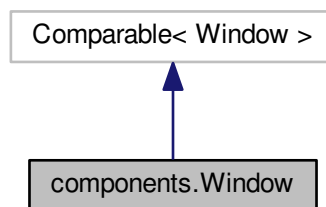
The documentation for this class was generated from the following file:

- /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/UnitRankCompare.java

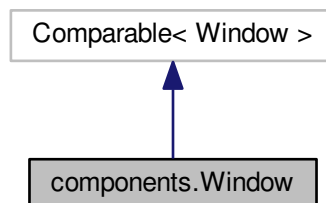
6.33 components.Window Class Reference

This class will store and provide functions for a single window within the model.

Inheritance diagram for components.Window:



Collaboration diagram for components.Window:



Public Member Functions

- [Window](#) (List< [Touch](#) > touches)
- [Window](#) ([Window](#) w)
copy constructor
- boolean [compare_with_token](#) (List< [Token](#) > tokens, [Window](#) other_window)
used for compairason of windows with a given token set.
- int [size](#) ()
returns the number of touches in the window
- List< [Touch](#) > [get_touch_list](#) ()
returns the window in the form of a touch list
- int [hashCode](#) ()
implement a hash function which returns the hash of the current window
- int [compareTo](#) ([Window](#) other_window)
compare this window to another window. Return negative if this window is less than the other window. Comparason is based on touches' pressure. Returns 0 if they are equal.
- String [toString](#) ()

6.33.1 Detailed Description

This class will store and provide functions for a single window within the model.

Windows are a list of n touches for a window of size n. When this class is used, The token which comes after the window is stored. It is this token which comes after the window which we compute the probability for.

6.33.2 Constructor & Destructor Documentation

6.33.2.1 `components.Window.Window (List< Touch > touches)`

6.33.2.2 `components.Window.Window (Window w)`

copy constructor

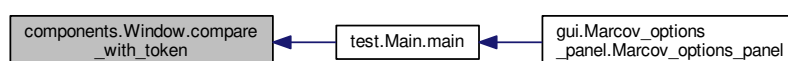
6.33.3 Member Function Documentation

6.33.3.1 `boolean components.Window.compare_with_token (List< Token > tokens, Window other_window)`

used for compairason of windows with a given token set.

return true if this window is equal to auth window.

Here is the caller graph for this function:



6.33.3.2 `int components.Window.compareTo (Window other_window)`

compare this window to another window. Return negative if this window is less than the other window. Comparason is based on touches' pressure. Returns 0 if they are equal.

Here is the call graph for this function:



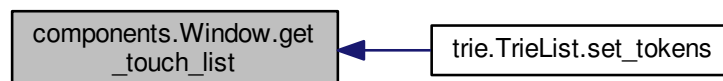
Here is the caller graph for this function:



6.33.3.3 `List<Touch> components.Window.get_touch_list ()`

returns the window in the form of a touch list

Here is the caller graph for this function:



6.33.3.4 `int components.Window.hashCode ()`

implement a hash function which returns the hash of the current window

6.33.3.5 `int components.Window.size ()`

returns the number of touches in the window

6.33.3.6 `String components.Window.toString ()`

The documentation for this class was generated from the following file:

- `/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/Window.java`

Chapter 7

File Documentation

7.1 /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/↔ Chain.java File Reference

Classes

- class [components.Chain](#)
Markov [Chain](#) built using keyboard tokens.

Packages

- package [components](#)

7.2 /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/↔ Distribution.java File Reference

Classes

- class [components.Distribution](#)
Used to compute and store max, min, std_deviation, and average for a list of [Touch](#).

Packages

- package [components](#)

7.3 /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/↔ Token.java File Reference

Classes

- class [components.Token](#)
This class represents a token within the model.
- enum [components.Token.Type](#)
specify the type of token we want to build

Packages

- package [components](#)

7.4 /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/↵ Touch.java File Reference

Classes

- class [components.Touch](#)
This class represents a touch event.

Packages

- package [components](#)

7.5 /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/components/↵ Window.java File Reference

Classes

- class [components.Window](#)
This class will store and provide functions for a single window within the model.

Packages

- package [components](#)

7.6 /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/data_analysis/↵ Model_compare.java File Reference

Classes

- class [data_analysis.Model_compare](#)
Analysis class used to compare and analyze data gathered from users.

Packages

- package [data_analysis](#)

7.7 [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/data_analysis/Model_compare_thread.java](#) File Reference

Reference

91

7.7 [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/data_analysis/Model_compare_thread.java](#) File Reference

Classes

- class [data_analysis.Model_compare_thread](#)
Compare Markov Chains on their own thread.

Packages

- package [data_analysis](#)

7.8 [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/data_analysis/Statistics.java](#) File Reference

Classes

- class [data_analysis.Statistics](#)
Generates statistics on results generated by model_compare.java.

Packages

- package [data_analysis](#)

7.9 [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/Marcov_console_panel.java](#) File Reference

Classes

- class [gui.Marcov_console_panel](#)
Displays messages printed to stdout.

Packages

- package [gui](#)

7.10 [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/Marcov_file_display_panel.java](#) File Reference

Classes

- class [gui.Marcov_file_display_panel](#)
Displays files relavant to test code.

Packages

- package [gui](#)

7.11 [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/Marcov_frame.java](#) File Reference

Classes

- class [gui.Marcov_frame](#)

Display frame to contain buttons for running test code and panels to view results.

Packages

- package [gui](#)

7.12 [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/Marcov_options_panel.java](#) File Reference

Classes

- class [gui.Marcov_options_panel](#)

Packages

- package [gui](#)

7.13 [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/gui/StartGUI.java](#) File Reference

Classes

- class [gui.StartGUI](#)

GUI useful for testing.

Packages

- package [gui](#)

7.14

/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/junit/Unit_CompareChainsRank.java

File Reference

93

7.14 /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/junit/Unit_↔

CompareChainsRank.java File Reference

Classes

- class [junit.Unit_CompareChainsRank](#)
goal is to test compare chains rank functionality

Packages

- package [junit](#)

7.15 /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/junit/Unit_↔

CompleteProbability.java File Reference

Classes

- class [junit.Unit_CompleteProbability](#)
unit test demonstrating how to compute probability

Packages

- package [junit](#)

7.16 /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/rank/Compare_↔

ChainsRank.java File Reference

Classes

- class [rank.CompareChainsRank](#)
Use PageRank algorithm (library) to compare chains.

Packages

- package [rank](#)

7.17 /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/rank/Complete_↔

Probability.java File Reference

Classes

- class [rank.CompleteProbability](#)
This class computes probability in a different way from what is contained in the Chain class.

Packages

- package [rank](#)

7.18 /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/runtime/ChainBuilder.java File Reference ↩↪

Classes

- class [runtime.ChainBuilder](#)
Wrapper around Chain used to make using the model easier in a real application.
- enum [runtime.ChainBuilder.State](#)
- enum [runtime.ChainBuilder.CompareMethod](#)

Packages

- package [runtime](#)

7.19 /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/runtime/CompareChains.java File Reference ↩↪

Classes

- class [runtime.CompareChains](#)
Use the compare method of the Chain class to determine an authentication probability between 0 and 1.

Packages

- package [runtime](#)

7.20 /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/runtime/Operation_thread.java File Reference ↩↪

Classes

- class [runtime.Operation_thread](#)
UNUSED.
- enum [runtime.Operation_thread.Computation](#)

Packages

- package [runtime](#)

7.21 /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/Main.java File Reference

Classes

- class [test.Main](#)
This class is used to test that the model is being built correctly.
- enum [test.Main.TestFiles.PressureAmount](#)
- enum [test.Main.TestFiles.Distribution](#)
- enum [test.Main.TestFiles.Concentration](#)

Packages

- package [test](#)

7.22 /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/Print_↵ model.java File Reference

Classes

- class [test.Print_model](#)
This class will print out the model constructed from the designated file.

Packages

- package [test](#)

7.23 /home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/UnitCompare↵ ChainsRank.java File Reference

Classes

- class [test.UnitCompareChainsRank](#)
JUnit test for testing PageRank version of compairason Not that the PageRank implementation is not currently functional.

Packages

- package [test](#)

7.24 [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/UnitRankCompare.java](#) File Reference

Classes

- class [test.UnitRankCompare](#)
Test the compairason with ranks.

Packages

- package [test](#)

7.25 [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/test/Utilities.java](#) File Reference

7.26 [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/trie/Trie.java](#) File Reference

Classes

- class [trie.Trie](#)
Implementation of Prefix Tree.
- class **trie.Trie.TrieNode**

Packages

- package [trie](#)

7.27 [/home/element/PUF/Keyboard/java_scripts/java_marcov_model/src/trie/TrieList.java](#) File Reference

Classes

- class [trie.TrieList](#)
Wrapper around [Trie](#) used to maintain an ordering among the stored elements.

Packages

- package [trie](#)

Index

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/components/Chain.java, 89

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/components/Distribution.↔
java, 89

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/components/Token.java, 89

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/components/Touch.java, 90

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/components/Window.java, 90

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/data_analysis/Model_↔
compare.java, 90

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/data_analysis/Model_↔
compare_thread.java, 91

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/data_analysis/Statistics.↔
java, 91

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/gui/Marcov_console_↔
panel.java, 91

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/gui/Marcov_file_display_↔
_panel.java, 91

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/gui/Marcov_frame.java, 92

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/gui/Marcov_options_↔
panel.java, 92

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/gui/StartGUI.java, 92

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/junit/Unit_Compare_↔
ChainsRank.java, 93

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/junit/Unit_Complete_↔
Probability.java, 93

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/rank/CompareChains_↔
Rank.java, 93

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/rank/CompleteProbability.↔
java, 93

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/runtime/ChainBuilder.java, 94

/home/element/PUF/Keyboard/java_scripts/java_↔
_marcov_model/src/runtime/Compare_↔
Chains.java, 94

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/runtime/Operation_↔
thread.java, 94

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/test/Main.java, 95

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/test/Print_model.java, 95

/home/element/PUF/Keyboard/java_scripts/java_↔
_marcov_model/src/test/UnitCompare_↔
ChainsRank.java, 95

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/test/UnitRankCompare.↔
java, 96

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/test/Utilities.java, 96

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/trie/Trie.java, 96

/home/element/PUF/Keyboard/java_scripts/java_↔
marcov_model/src/trie/TrieList.java, 96

ABNORMAL

test::Main::TestFiles::Distribution, 37

add

trie::TrieList, 75

add_touch

components::Chain, 14

add_touch_list

components::Chain, 14

addAll

trie::TrieList, 75

auth_chain

runtime::CompareChains, 30

authenticate

runtime::ChainBuilder, 24

authentication_accuracy

data_analysis::Statistics, 59

authentication_probability

runtime::CompareChains, 30

average_authentication_probability

data_analysis::Model_compare_thread, 52

best_authentication_percentage

data_analysis::Statistics, 59

- build_chain_from_csv
 - runtime::ChainBuilder, 24
- Chain
 - components::Chain, 14
- ChainBuilder
 - runtime::ChainBuilder, 24
- clear
 - trie::Trie, 72
 - trie::TrieList, 76
- close
 - gui::Marcov_frame, 46
- combined
 - components::Token::Type, 79
- compare_to
 - components::Chain, 15
- compare_with_token
 - components::Touch, 68
 - components::Window, 87
- CompareChains
 - runtime::CompareChains, 28
- CompareChainsRank
 - rank::CompareChainsRank, 32
- compareTo
 - components::Touch, 69
 - components::Window, 87
- complete
 - runtime::CompareChains, 30
- CompleteProbability
 - rank::CompleteProbability, 34
- components, 9
- components.Chain, 13
- components.Distribution, 37
- components.Token, 62
- components.Token.Type, 79
- components.Touch, 67
- components.Window, 86
- components::Chain
 - add_touch, 14
 - add_touch_list, 14
 - Chain, 14
 - compare_to, 15
 - compute_uncomputed, 16
 - get_distribution, 16
 - get_key_distribution, 17
 - get_model_size, 17
 - get_threshold, 17
 - get_token, 18
 - get_tokens, 18
 - get_touch_probability, 18
 - get_touches, 19
 - get_window, 19
 - get_windows, 20
 - is_touch_in_key_distribution, 20
 - output_to_csv, 21
 - reset, 21
 - set_distribution, 22
 - toString, 22
- components::Distribution
 - Distribution, 38
 - equals, 38
 - get_average, 39
 - get_keycode, 39
 - get_max, 39
 - get_min, 40
 - get_standard_deviation, 40
 - update, 40
- components::Token
 - contains, 64
 - equals, 64
 - get_acceptable_wildcards, 65
 - get_max, 65
 - get_min, 65
 - get_total_wildcards, 66
 - increment_high_wildcards, 66
 - increment_low_wildcards, 66
 - is_high_wildcard, 66
 - is_low_wildcard, 66
 - Token, 63, 64
- components::Token::Type
 - combined, 79
 - keycode_mu, 79
 - linear, 79
- components::Touch
 - compare_with_token, 68
 - compareTo, 69
 - get_key, 69
 - get_pressure, 69
 - get_probability, 70
 - get_timestamp, 70
 - hashCode, 70
 - set_probability, 70
 - toString, 71
 - Touch, 68
- components::Window
 - compare_with_token, 87
 - compareTo, 87
 - get_touch_list, 88
 - hashCode, 88
 - size, 88
 - toString, 88
 - Window, 87
- compute_probability
 - rank::CompleteProbability, 34
- compute_uncomputed
 - components::Chain, 16
- Concentration
 - test::Main::TestFiles::Concentration, 36
- contains
 - components::Token, 64
- DISTRIBUTION
 - runtime::Operation_thread::Computation, 35
- data_analysis, 9
- data_analysis.Model_compare, 48
- data_analysis.Model_compare_thread, 50
- data_analysis.Statistics, 58
- data_analysis::Model_compare

- main, [49](#)
- data_analysis::Model_compare_thread
 - average_authentication_probability, [52](#)
 - get_auth_data_path, [51](#)
 - get_auth_model_size, [51](#)
 - get_auth_probability_list, [51](#)
 - get_base_data_path, [51](#)
 - get_base_model_size, [51](#)
 - get_threshold, [51](#)
 - get_token_size, [51](#)
 - get_window_size, [51](#)
 - max_authentication_probability, [52](#)
 - min_authentication_probability, [52](#)
 - Model_compare_thread, [51](#)
 - run, [51](#)
- data_analysis::Statistics
 - authentication_accuracy, [59](#)
 - best_authentication_percentage, [59](#)
 - equal_false_positive_negative_authentication_↔
percentage, [59](#)
 - false_negative_percentage, [60](#)
 - false_positive_percentage, [60](#)
 - main, [61](#)
 - minimize_false_positive_authentication_percentage,
[61](#)
- Distribution
 - components::Distribution, [38](#)
 - test::Main::TestFiles::Distribution, [37](#)
- equal_false_positive_negative_authentication_↔
percentage
 - data_analysis::Statistics, [59](#)
- equals
 - components::Distribution, [38](#)
 - components::Token, [64](#)
- exit
 - gui::StartGUI, [57](#)
- false_negative_percentage
 - data_analysis::Statistics, [60](#)
- false_positive_percentage
 - data_analysis::Statistics, [60](#)
- get_acceptable_wildcards
 - components::Token, [65](#)
- get_auth_complete
 - runtime::CompareChains, [29](#)
- get_auth_data_path
 - data_analysis::Model_compare_thread, [51](#)
- get_auth_model_size
 - data_analysis::Model_compare_thread, [51](#)
- get_auth_probability
 - runtime::CompareChains, [29](#)
- get_auth_probability_list
 - data_analysis::Model_compare_thread, [51](#)
- get_auth_result
 - runtime::CompareChains, [29](#)
- get_authenticate_state
 - runtime::ChainBuilder, [25](#)
- get_authenticate_thread
 - runtime::ChainBuilder, [25](#)
- get_average
 - components::Distribution, [39](#)
- get_base_data_path
 - data_analysis::Model_compare_thread, [51](#)
- get_base_model_size
 - data_analysis::Model_compare_thread, [51](#)
- get_distribution
 - components::Chain, [16](#)
- get_identifier
 - test::Main::TestFiles::Concentration, [36](#)
 - test::Main::TestFiles::Distribution, [37](#)
 - test::Main::TestFiles::PressureAmount, [55](#)
- get_index_list
 - trie::Trie, [72](#)
- get_key
 - components::Touch, [69](#)
- get_key_distribution
 - components::Chain, [17](#)
- get_keycode
 - components::Distribution, [39](#)
- get_max
 - components::Distribution, [39](#)
 - components::Token, [65](#)
- get_min
 - components::Distribution, [40](#)
 - components::Token, [65](#)
- get_model_size
 - components::Chain, [17](#)
- get_pressure
 - components::Touch, [69](#)
- get_probability
 - components::Touch, [70](#)
- get_standard_deviation
 - components::Distribution, [40](#)
- get_threshold
 - components::Chain, [17](#)
 - data_analysis::Model_compare_thread, [51](#)
- get_timestamp
 - components::Touch, [70](#)
- get_token
 - components::Chain, [18](#)
- get_token_size
 - data_analysis::Model_compare_thread, [51](#)
- get_tokens
 - components::Chain, [18](#)
- get_total_wildcards
 - components::Token, [66](#)
- get_touch_list
 - components::Window, [88](#)
- get_touch_probability
 - components::Chain, [18](#)
- get_touches
 - components::Chain, [19](#)
- get_value
 - test::Main::TestFiles::Concentration, [36](#)
 - test::Main::TestFiles::Distribution, [37](#)

- test::Main::TestFiles::PressureAmount, 55
- get_window
 - components::Chain, 19
- get_window_size
 - data_analysis::Model_compare_thread, 51
- get_windows
 - components::Chain, 20
- gui, 10
- gui.Marcov_console_panel, 43
- gui.Marcov_file_display_panel, 44
- gui.Marcov_frame, 45
- gui.Marcov_options_panel, 46
- gui.StartGUI, 56
- gui::Marcov_console_panel
 - Marcov_console_panel, 44
- gui::Marcov_file_display_panel
 - Marcov_file_display_panel, 45
- gui::Marcov_frame
 - close, 46
 - Marcov_frame, 46
- gui::Marcov_options_panel
 - Marcov_options_panel, 47
- gui::StartGUI
 - exit, 57
 - main, 57
- HIGH
 - test::Main::TestFiles::Concentration, 36
 - test::Main::TestFiles::PressureAmount, 55
- handle_touch
 - runtime::ChainBuilder, 26
- hashCode
 - components::Touch, 70
 - components::Window, 88
- IN_PROGRESS
 - runtime::ChainBuilder::State, 58
- increment_high_wildcards
 - components::Token, 66
- increment_low_wildcards
 - components::Token, 66
- init
 - junit::Unit_CompareChainsRank, 80
 - junit::Unit_CompleteProbability, 81
 - test::UnitCompareChainsRank, 83
 - test::UnitRankCompare, 85
- insertString
 - trie::Trie, 72
- is_authentic
 - runtime::CompareChains, 30
- is_high_wildcard
 - components::Token, 66
- is_low_wildcard
 - components::Token, 66
- is_touch_in_key_distribution
 - components::Chain, 20
- junit, 10
- junit.Unit_CompareChainsRank, 79
- junit.Unit_CompleteProbability, 80
- junit::Unit_CompareChainsRank
 - init, 80
 - test_authentication_probability, 80
- junit::Unit_CompleteProbability
 - init, 81
 - test_replica_distribution, 81
- KEY_DISTRIBUTION
 - runtime::Operation_thread::Computation, 35
- keycode_mu
 - components::Token::Type, 79
- LOW
 - test::Main::TestFiles::Concentration, 36
 - test::Main::TestFiles::PressureAmount, 55
- linear
 - components::Token::Type, 79
- MEDIUM
 - test::Main::TestFiles::Concentration, 36
 - test::Main::TestFiles::PressureAmount, 55
- main
 - data_analysis::Model_compare, 49
 - data_analysis::Statistics, 61
 - gui::StartGUI, 57
 - test::Main, 42
 - test::Print_model, 56
- Marcov_console_panel
 - gui::Marcov_console_panel, 44
- Marcov_file_display_panel
 - gui::Marcov_file_display_panel, 45
- Marcov_frame
 - gui::Marcov_frame, 46
- Marcov_options_panel
 - gui::Marcov_options_panel, 47
- max_authentication_probability
 - data_analysis::Model_compare_thread, 52
- min_authentication_probability
 - data_analysis::Model_compare_thread, 52
- minimize_false_positive_authentication_percentage
 - data_analysis::Statistics, 61
- Model_compare_thread
 - data_analysis::Model_compare_thread, 51
- NORMAL
 - test::Main::TestFiles::Distribution, 37
- occurrence_count
 - trie::Trie, 73
 - trie::TrieList, 76
- Operation_thread
 - runtime::Operation_thread, 54
- output_to_csv
 - components::Chain, 21
- PROBABILITY_VECTOR_DIFFERENCE
 - runtime::ChainBuilder::CompareMethod, 33
- PROBABILITY
 - runtime::Operation_thread::Computation, 35

- parse_csv
 - runtime::ChainBuilder, 26
- PressureAmount
 - test::Main::TestFiles::PressureAmount, 55
- printSorted
 - trie::Trie, 73
- RANDOM
 - test::Main::TestFiles::Distribution, 37
- rank, 10
- rank.CompareChainsRank, 31
- rank.CompleteProbability, 33
- rank::CompareChainsRank
 - CompareChainsRank, 32
 - run, 32
- rank::CompleteProbability
 - CompleteProbability, 34
 - compute_probability, 34
- remove
 - trie::TrieList, 77
- removeAll
 - trie::TrieList, 77
- reset
 - components::Chain, 21
- retainAll
 - trie::TrieList, 77
- run
 - data_analysis::Model_compare_thread, 51
 - rank::CompareChainsRank, 32
 - runtime::CompareChains, 29
 - runtime::Operation_thread, 54
- runtime, 10
- runtime.ChainBuilder, 23
- runtime.ChainBuilder.CompareMethod, 33
- runtime.ChainBuilder.State, 58
- runtime.CompareChains, 27
- runtime.Operation_thread, 52
- runtime.Operation_thread.Computation, 35
- runtime::ChainBuilder
 - authenticate, 24
 - build_chain_from_csv, 24
 - ChainBuilder, 24
 - get_authenticate_state, 25
 - get_authenticate_thread, 25
 - handle_touch, 26
 - parse_csv, 26
- runtime::ChainBuilder::CompareMethod
 - PROBABILITY_VECTOR_DIFFERENCE, 33
- runtime::ChainBuilder::State
 - IN_PROGRESS, 58
 - SUCCESS, 58
- runtime::CompareChains
 - auth_chain, 30
 - authentication_probability, 30
 - CompareChains, 28
 - complete, 30
 - get_auth_complete, 29
 - get_auth_probability, 29
 - get_auth_result, 29
 - is_authentic, 30
 - run, 29
 - user_chain, 30
- runtime::Operation_thread
 - Operation_thread, 54
 - run, 54
- runtime::Operation_thread::Computation
 - DISTRIBUTION, 35
 - KEY_DISTRIBUTION, 35
 - PROBABILITY, 35
 - TOKEN, 35
 - WINDOW, 35
- SUCCESS
 - runtime::ChainBuilder::State, 58
- set
 - trie::TrieList, 77
- set_distribution
 - components::Chain, 22
- set_probability
 - components::Touch, 70
- set_tokens
 - trie::TrieList, 78
- size
 - components::Window, 88
- successor_count
 - trie::TrieList, 78
- TOKEN
 - runtime::Operation_thread::Computation, 35
- test, 11
 - test::UnitCompareChainsRank, 83
 - test::UnitRankCompare, 85
- test.Main, 41
- test.Main.TestFiles.Concentration, 35
- test.Main.TestFiles.Distribution, 36
- test.Main.TestFiles.PressureAmount, 54
- test.Print_model, 55
- test.UnitCompareChainsRank, 82
- test.UnitRankCompare, 84
- test::Main
 - main, 42
- test::Main::TestFiles::Concentration
 - Concentration, 36
 - get_identifier, 36
 - get_value, 36
 - HIGH, 36
 - LOW, 36
 - MEDIUM, 36
 - toString, 36
- test::Main::TestFiles::Distribution
 - ABNORMAL, 37
 - Distribution, 37
 - get_identifier, 37
 - get_value, 37
 - NORMAL, 37
 - RANDOM, 37
 - toString, 37
- test::Main::TestFiles::PressureAmount

- get_identifier, 55
 - get_value, 55
 - HIGH, 55
 - LOW, 55
 - MEDIUM, 55
 - PressureAmount, 55
 - toString, 55
- test::Print_model
 - main, 56
- test::UnitCompareChainsRank
 - init, 83
 - test, 83
 - test_chain_to_graph, 83
 - test_touch_index, 83
 - test_touch_window, 83
- test::UnitRankCompare
 - init, 85
 - test, 85
 - test_auth_probability, 85
 - test_compare_correct, 85
- test_auth_probability
 - test::UnitRankCompare, 85
- test_authentication_probability
 - junit::Unit_CompareChainsRank, 80
- test_chain_to_graph
 - test::UnitCompareChainsRank, 83
- test_compare_correct
 - test::UnitRankCompare, 85
- test_replica_distribution
 - junit::Unit_CompleteProbability, 81
- test_touch_index
 - test::UnitCompareChainsRank, 83
- test_touch_window
 - test::UnitCompareChainsRank, 83
- toString
 - components::Chain, 22
 - components::Touch, 71
 - components::Window, 88
 - test::Main::TestFiles::Concentration, 36
 - test::Main::TestFiles::Distribution, 37
 - test::Main::TestFiles::PressureAmount, 55
- Token
 - components::Token, 63, 64
- Touch
 - components::Touch, 68
- Trie
 - trie::Trie, 72
- trie, 11
- trie.Trie, 71
- trie.TrieList, 73
- trie::Trie
 - clear, 72
 - get_index_list, 72
 - insertString, 72
 - occurrence_count, 73
 - printSorted, 73
 - Trie, 72
- trie::TrieList
 - add, 75
 - addAll, 75
 - clear, 76
 - occurrence_count, 76
 - remove, 77
 - removeAll, 77
 - retainAll, 77
 - set, 77
 - set_tokens, 78
 - successor_count, 78
 - TrieList, 75
- TrieList
 - trie::TrieList, 75
- update
 - components::Distribution, 40
- user_chain
 - runtime::CompareChains, 30
- WINDOW
 - runtime::Operation_thread::Computation, 35
- Window
 - components::Window, 87