

**BRisa Qt**  
r993

Generated by Doxygen 1.6.1

Wed Mar 24 08:38:10 2010



# Contents

<b>1 Namespace Index</b>	<b>1</b>
1.1 Namespace List . . . . .	1
<b>2 Class Index</b>	<b>3</b>
2.1 Class Hierarchy . . . . .	3
<b>3 Class Index</b>	<b>5</b>
3.1 Class List . . . . .	5
<b>4 File Index</b>	<b>7</b>
4.1 File List . . . . .	7
<b>5 Namespace Documentation</b>	<b>9</b>
5.1 BrisaCore Namespace Reference . . . . .	9
5.2 BrisaUpnp Namespace Reference . . . . .	10
5.2.1 Enumeration Type Documentation . . . . .	11
5.2.1.1 SaxParserState . . . . .	11
5.2.1.2 saxParserState . . . . .	12
<b>6 Class Documentation</b>	<b>15</b>
6.1 BrisaCore::BrisaConfigurationManager Class Reference . . . . .	15
6.1.1 Detailed Description . . . . .	16
6.1.2 Constructor & Destructor Documentation . . . . .	16
6.1.2.1 BrisaConfigurationManager . . . . .	16
6.1.3 Member Function Documentation . . . . .	16
6.1.3.1 getDirectAccess . . . . .	16
6.1.3.2 getParameter . . . . .	17
6.1.3.3 getSectionNames . . . . .	17
6.1.3.4 items . . . . .	17
6.1.3.5 removeSection . . . . .	17

6.1.3.6	save . . . . .	17
6.1.3.7	setDirectAccess . . . . .	18
6.1.3.8	setParameter . . . . .	18
6.1.3.9	update . . . . .	18
6.2	BrisaCore::BrisaWebFile Class Reference . . . . .	19
6.2.1	Detailed Description . . . . .	19
6.2.2	Constructor & Destructor Documentation . . . . .	19
6.2.2.1	BrisaWebFile . . . . .	19
6.2.2.2	~BrisaWebFile . . . . .	19
6.2.3	Member Function Documentation . . . . .	19
6.2.3.1	pageRequestedEvent . . . . .	19
6.3	BrisaCore::BrisaWebserver Class Reference . . . . .	20
6.3.1	Detailed Description . . . . .	21
6.3.2	Constructor & Destructor Documentation . . . . .	21
6.3.2.1	BrisaWebserver . . . . .	21
6.3.2.2	~BrisaWebserver . . . . .	21
6.3.3	Member Function Documentation . . . . .	21
6.3.3.1	addService . . . . .	21
6.3.3.2	incomingRequest . . . . .	21
6.3.3.3	newSession . . . . .	21
6.3.3.4	publishFile . . . . .	21
6.4	BrisaCore::BrisaWebService Class Reference . . . . .	23
6.4.1	Detailed Description . . . . .	24
6.4.2	Constructor & Destructor Documentation . . . . .	24
6.4.2.1	BrisaWebService . . . . .	24
6.4.2.2	~BrisaWebService . . . . .	24
6.4.3	Member Function Documentation . . . . .	25
6.4.3.1	genericRequestReceived . . . . .	25
6.4.3.2	genericRequestReceived . . . . .	25
6.4.3.3	pageRequestedEvent . . . . .	25
6.4.3.4	respond . . . . .	25
6.4.3.5	respond . . . . .	25
6.4.3.6	respond . . . . .	25
6.4.3.7	respond . . . . .	25
6.5	BrisaCore::BrisaServiceProvider Class Reference . . . . .	26
6.5.1	Detailed Description . . . . .	27

---

6.5.2	Constructor & Destructor Documentation . . . . .	27
6.5.2.1	BrisaWebServiceProvider . . . . .	27
6.5.2.2	~BrisaWebServiceProvider . . . . .	27
6.5.3	Member Function Documentation . . . . .	27
6.5.3.1	addContent . . . . .	27
6.5.3.2	addFile . . . . .	27
6.5.3.3	indexRequested . . . . .	27
6.5.3.4	pageRequestedEvent . . . . .	27
6.6	BrisaCore::BrisaWebStaticContent Class Reference . . . . .	29
6.6.1	Detailed Description . . . . .	29
6.6.2	Constructor & Destructor Documentation . . . . .	29
6.6.2.1	BrisaWebStaticContent . . . . .	29
6.6.2.2	~BrisaWebStaticContent . . . . .	29
6.6.3	Member Function Documentation . . . . .	29
6.6.3.1	index . . . . .	29
6.7	BrisaUpnp::BrisaAbstractEventSubscription Class Reference . . . . .	30
6.7.1	Constructor & Destructor Documentation . . . . .	31
6.7.1.1	BrisaAbstractEventSubscription . . . . .	31
6.7.2	Member Function Documentation . . . . .	31
6.7.2.1	getCallbackUrls . . . . .	31
6.7.2.2	getNextSeq . . . . .	31
6.7.2.3	getSid . . . . .	31
6.7.2.4	getUrl . . . . .	32
6.7.2.5	hasExpired . . . . .	32
6.7.2.6	renew . . . . .	32
6.7.3	Member Data Documentation . . . . .	32
6.7.3.1	CALLBACK_URLS . . . . .	32
6.7.3.2	date . . . . .	32
6.7.3.3	firstMessageSent . . . . .	32
6.7.3.4	lastSeq . . . . .	32
6.7.3.5	SID . . . . .	32
6.7.3.6	timeout . . . . .	32
6.8	BrisaUpnp::BrisaAbstractService Class Reference . . . . .	33
6.8.1	Member Enumeration Documentation . . . . .	34
6.8.1.1	xmlTags . . . . .	34
6.8.2	Constructor & Destructor Documentation . . . . .	35

---

6.8.2.1	BrisaAbstractService . . . . .	35
6.8.2.2	BrisaAbstractService . . . . .	35
6.8.2.3	BrisaAbstractService . . . . .	35
6.8.2.4	~BrisaAbstractService . . . . .	35
6.8.3	Member Function Documentation . . . . .	35
6.8.3.1	addAction . . . . .	35
6.8.3.2	addAction . . . . .	35
6.8.3.3	addStateVariable . . . . .	35
6.8.3.4	addStateVariable . . . . .	35
6.8.3.5	call . . . . .	35
6.8.3.6	clear . . . . .	35
6.8.3.7	getAction . . . . .	35
6.8.3.8	getActionList . . . . .	36
6.8.3.9	getAttribute . . . . .	36
6.8.3.10	getStateVariableList . . . . .	36
6.8.3.11	requestFinished . . . . .	36
6.8.3.12	setAttribute . . . . .	36
6.8.4	Member Data Documentation . . . . .	36
6.8.4.1	actionList . . . . .	36
6.8.4.2	controlUrl . . . . .	36
6.8.4.3	eventSubUrl . . . . .	36
6.8.4.4	fileAddress . . . . .	36
6.8.4.5	host . . . . .	36
6.8.4.6	http . . . . .	37
6.8.4.7	major . . . . .	37
6.8.4.8	minor . . . . .	37
6.8.4.9	port . . . . .	37
6.8.4.10	scpdUrl . . . . .	37
6.8.4.11	serviceId . . . . .	37
6.8.4.12	serviceType . . . . .	37
6.8.4.13	stateVariableList . . . . .	37
6.9	BrisaUpnp::BrisaAction Class Reference . . . . .	38
6.9.1	Detailed Description . . . . .	39
6.9.2	Constructor & Destructor Documentation . . . . .	40
6.9.2.1	BrisaAction . . . . .	40
6.9.2.2	BrisaAction . . . . .	40

---

6.9.2.3	~BrisaAction	40
6.9.3	Member Function Documentation	40
6.9.3.1	addArgument	40
6.9.3.2	addArgument	40
6.9.3.3	addArguments	40
6.9.3.4	call	40
6.9.3.5	clearArgumentList	40
6.9.3.6	getArgumentList	41
6.9.3.7	getName	41
6.9.3.8	getService	41
6.9.3.9	getStateVariable	41
6.9.3.10	run	41
6.9.3.11	setName	41
6.9.3.12	setService	41
6.10	BrisaUpnp::BrisaActionXmlParser Class Reference	42
6.10.1	Detailed Description	42
6.10.2	Constructor & Destructor Documentation	42
6.10.2.1	BrisaActionXmlParser	42
6.10.2.2	~BrisaActionXmlParser	42
6.10.3	Member Function Documentation	43
6.10.3.1	parseElement	43
6.10.3.2	parseSOAP	43
6.10.3.3	setXmlContent	43
6.10.4	Member Data Documentation	43
6.10.4.1	args	43
6.10.4.2	method	43
6.10.4.3	serviceType	43
6.11	BrisaUpnp::BrisaArgument Class Reference	44
6.11.1	Member Enumeration Documentation	44
6.11.1.1	xmlArgument	44
6.11.2	Constructor & Destructor Documentation	44
6.11.2.1	BrisaArgument	44
6.11.3	Member Function Documentation	44
6.11.3.1	clear	44
6.11.3.2	getAttribute	44
6.11.3.3	setAttribute	44

6.12 BrisaUpnp::BrisaControlPoint Class Reference . . . . .	45
6.12.1 Detailed Description . . . . .	47
6.12.2 Constructor & Destructor Documentation . . . . .	47
6.12.2.1 BrisaControlPoint . . . . .	47
6.12.2.2 ~BrisaControlPoint . . . . .	48
6.12.3 Member Function Documentation . . . . .	48
6.12.3.1 deviceFound . . . . .	48
6.12.3.2 deviceGone . . . . .	48
6.12.3.3 discover . . . . .	48
6.12.3.4 getSubscriptionProxy . . . . .	48
6.12.3.5 isRunning . . . . .	48
6.12.3.6 start . . . . .	48
6.12.3.7 stop . . . . .	49
6.13 BrisaUpnp::BrisaControlPointDevice Class Reference . . . . .	50
6.13.1 Detailed Description . . . . .	51
6.13.2 Member Enumeration Documentation . . . . .	51
6.13.2.1 xmlTags . . . . .	51
6.13.3 Constructor & Destructor Documentation . . . . .	52
6.13.3.1 BrisaControlPointDevice . . . . .	52
6.13.3.2 BrisaControlPointDevice . . . . .	52
6.13.3.3 BrisaControlPointDevice . . . . .	52
6.13.3.4 BrisaControlPointDevice . . . . .	53
6.13.3.5 ~BrisaControlPointDevice . . . . .	53
6.13.4 Member Function Documentation . . . . .	53
6.13.4.1 addDevice . . . . .	53
6.13.4.2 addIcon . . . . .	53
6.13.4.3 addService . . . . .	53
6.13.4.4 clear . . . . .	53
6.13.4.5 getAttribute . . . . .	53
6.13.4.6 getEmbeddedDeviceList . . . . .	53
6.13.4.7 getIconList . . . . .	54
6.13.4.8 getServiceById . . . . .	54
6.13.4.9 getServiceByType . . . . .	54
6.13.4.10 getServiceList . . . . .	54
6.13.4.11 setAttribute . . . . .	54
6.14 BrisaUpnp::BrisaControlPointService Class Reference . . . . .	55

6.14.1	Detailed Description	57
6.14.2	Constructor & Destructor Documentation	57
6.14.2.1	BrisaControlPointService	57
6.14.2.2	BrisaControlPointService	57
6.14.2.3	BrisaControlPointService	57
6.14.3	Member Function Documentation	57
6.14.3.1	call	57
6.14.3.2	parseFromXml	57
6.15	BrisaUpnp::BrisaDevice Class Reference	58
6.15.1	Detailed Description	60
6.15.2	Member Enumeration Documentation	61
6.15.2.1	xmlTags	61
6.15.3	Constructor & Destructor Documentation	61
6.15.3.1	BrisaDevice	61
6.15.3.2	BrisaDevice	61
6.15.3.3	BrisaDevice	62
6.15.3.4	~BrisaDevice	62
6.15.4	Member Function Documentation	62
6.15.4.1	addEmbeddedDevice	62
6.15.4.2	addEmbeddedDevice	62
6.15.4.3	addIcon	62
6.15.4.4	addService	62
6.15.4.5	addService	62
6.15.4.6	clear	63
6.15.4.7	doByeBye	63
6.15.4.8	doNotify	63
6.15.4.9	getAttribute	63
6.15.4.10	getEmbeddedDeviceList	63
6.15.4.11	getIconList	63
6.15.4.12	getServiceById	63
6.15.4.13	getServiceByType	64
6.15.4.14	getServiceList	64
6.15.4.15	operator=	64
6.15.4.16	respondMSearch	64
6.15.4.17	setAttribute	64
6.15.4.18	start	64

6.15.4.19	stop	64
6.16	BrisaUpnp::BrisaDeviceParserContext Class Reference	67
6.16.1	Constructor & Destructor Documentation	68
6.16.1.1	BrisaDeviceParserContext	68
6.16.2	Member Function Documentation	68
6.16.2.1	getDevice	68
6.16.2.2	getIcon	68
6.16.2.3	getParent	68
6.16.2.4	getService	68
6.16.2.5	hasParent	69
6.16.2.6	setDevice	69
6.16.2.7	setIcon	69
6.16.2.8	setService	69
6.16.3	Member Data Documentation	69
6.16.3.1	state	69
6.16.3.2	stateSkip	69
6.17	BrisaUpnp::BrisaDeviceXMLHandler Class Reference	70
6.17.1	Member Function Documentation	70
6.17.1.1	xmlGenerator	70
6.18	BrisaUpnp::BrisaDeviceXMLHandlerCP Class Reference	71
6.18.1	Detailed Description	73
6.18.2	Member Function Documentation	73
6.18.2.1	characters	73
6.18.2.2	endElement	73
6.18.2.3	parseDevice	73
6.18.2.4	startElement	74
6.19	BrisaUpnp::BrisaEventController Class Reference	75
6.19.1	Constructor & Destructor Documentation	77
6.19.1.1	BrisaEventController	77
6.19.1.2	~BrisaEventController	77
6.19.2	Member Function Documentation	77
6.19.2.1	parseGenericRequest	77
6.19.2.2	subscribe	77
6.19.2.3	unsubscribe	77
6.19.2.4	variableChanged	78
6.20	BrisaUpnp::BrisaEventMessage Class Reference	79

6.20.1	Constructor & Destructor Documentation . . . . .	80
6.20.1.1	BrisaEventMessage . . . . .	80
6.20.2	Member Function Documentation . . . . .	80
6.20.2.1	getMessageBody . . . . .	80
6.20.2.2	getMessageHeader . . . . .	80
6.21	BrisaUpnp::BrisaEventProxy Class Reference . . . . .	81
6.21.1	Detailed Description . . . . .	83
6.21.2	Constructor & Destructor Documentation . . . . .	83
6.21.2.1	~BrisaEventProxy . . . . .	83
6.21.3	Member Function Documentation . . . . .	83
6.21.3.1	eventNotification . . . . .	83
6.21.3.2	getId . . . . .	83
6.21.3.3	renew . . . . .	84
6.21.3.4	subscribe . . . . .	84
6.21.3.5	unsubscribe . . . . .	84
6.21.4	Friends And Related Function Documentation . . . . .	84
6.21.4.1	BrisaControlPoint . . . . .	84
6.22	BrisaUpnp::BrisaEventSubscription Class Reference . . . . .	85
6.22.1	Constructor & Destructor Documentation . . . . .	87
6.22.1.1	BrisaEventSubscription . . . . .	87
6.22.2	Member Function Documentation . . . . .	87
6.22.2.1	getAcceptSubscriptionResponse . . . . .	87
6.22.2.2	getAcceptUnsubscriptionResponse . . . . .	87
6.22.2.3	renew . . . . .	87
6.23	BrisaUpnp::BrisaIcon Class Reference . . . . .	88
6.23.1	Member Enumeration Documentation . . . . .	88
6.23.1.1	xmlIconTags . . . . .	88
6.23.2	Constructor & Destructor Documentation . . . . .	88
6.23.2.1	BrisaIcon . . . . .	88
6.23.3	Member Function Documentation . . . . .	88
6.23.3.1	clear . . . . .	88
6.23.3.2	getAttribute . . . . .	88
6.23.3.3	setAttribute . . . . .	88
6.24	BrisaUpnp::BrisaMSearchClientCP Class Reference . . . . .	89
6.24.1	Detailed Description . . . . .	89
6.24.2	Constructor & Destructor Documentation . . . . .	89

6.24.2.1	BrisaMSearchClientCP	89
6.24.2.2	~BrisaMSearchClientCP	89
6.24.3	Member Function Documentation	89
6.24.3.1	discover	89
6.24.3.2	doubleDiscover	90
6.24.3.3	isRunning	90
6.24.3.4	msearchResponseReceived	90
6.24.3.5	start	90
6.24.3.6	stop	90
6.25	BrisaUpnp::BrisaService Class Reference	91
6.25.1	Detailed Description	93
6.25.2	Constructor & Destructor Documentation	93
6.25.2.1	BrisaService	93
6.25.2.2	BrisaService	93
6.25.2.3	BrisaService	93
6.25.2.4	~BrisaService	93
6.25.3	Member Function Documentation	94
6.25.3.1	buildWebServiceTree	94
6.25.3.2	getVariable	94
6.25.3.3	getWebService	94
6.25.3.4	parseGenericRequest	94
6.25.3.5	setDescriptionFile	94
6.26	BrisaUpnp::BrisaServiceFetcher Class Reference	95
6.26.1	Constructor & Destructor Documentation	96
6.26.1.1	BrisaServiceFetcher	96
6.26.1.2	~BrisaServiceFetcher	96
6.26.2	Member Function Documentation	96
6.26.2.1	fetch	96
6.26.2.2	fetchFinished	96
6.27	BrisaUpnp::BrisaServiceParserContext Class Reference	98
6.27.1	Constructor & Destructor Documentation	99
6.27.1.1	BrisaServiceParserContext	99
6.27.2	Member Function Documentation	99
6.27.2.1	getAction	99
6.27.2.2	getArgument	99
6.27.2.3	getParent	99

6.27.2.4	getService . . . . .	99
6.27.2.5	getStateVariable . . . . .	100
6.27.2.6	hasParent . . . . .	100
6.27.2.7	setAction . . . . .	100
6.27.2.8	setArgument . . . . .	100
6.27.2.9	setService . . . . .	100
6.27.2.10	setStateVariable . . . . .	100
6.27.3	Member Data Documentation . . . . .	100
6.27.3.1	state . . . . .	100
6.27.3.2	stateSkip . . . . .	100
6.28	BrisaUpnp::BrisaServiceXMLHandler Class Reference . . . . .	101
6.28.1	Member Function Documentation . . . . .	103
6.28.1.1	characters . . . . .	103
6.28.1.2	endElement . . . . .	103
6.28.1.3	parseService . . . . .	103
6.28.1.4	startElement . . . . .	103
6.29	BrisaUpnp::BrisaSSDPClient Class Reference . . . . .	104
6.29.1	Detailed Description . . . . .	104
6.29.2	Constructor & Destructor Documentation . . . . .	104
6.29.2.1	BrisaSSDPClient . . . . .	104
6.29.2.2	~BrisaSSDPClient . . . . .	105
6.29.3	Member Function Documentation . . . . .	105
6.29.3.1	isRunning . . . . .	105
6.29.3.2	newDeviceEvent . . . . .	105
6.29.3.3	removedDeviceEvent . . . . .	105
6.29.3.4	start . . . . .	105
6.29.3.5	stop . . . . .	105
6.30	BrisaUpnp::BrisaSSDPServer Class Reference . . . . .	106
6.30.1	Detailed Description . . . . .	107
6.30.2	Constructor & Destructor Documentation . . . . .	107
6.30.2.1	BrisaSSDPServer . . . . .	107
6.30.2.2	~BrisaSSDPServer . . . . .	107
6.30.3	Member Function Documentation . . . . .	107
6.30.3.1	doByeBye . . . . .	107
6.30.3.2	doNotify . . . . .	107
6.30.3.3	isRunning . . . . .	107

6.30.3.4	msearchRequestReceived . . . . .	108
6.30.3.5	respondMSearch . . . . .	108
6.30.3.6	start . . . . .	108
6.30.3.7	stop . . . . .	108
6.31	BrisaUpnp::BrisaStateVariable Class Reference . . . . .	109
6.31.1	Detailed Description . . . . .	110
6.31.2	Member Enumeration Documentation . . . . .	110
6.31.2.1	BrisaStateVariableAttribute . . . . .	110
6.31.3	Constructor & Destructor Documentation . . . . .	110
6.31.3.1	BrisaStateVariable . . . . .	110
6.31.3.2	BrisaStateVariable . . . . .	110
6.31.4	Member Function Documentation . . . . .	110
6.31.4.1	addAllowedValue . . . . .	110
6.31.4.2	changed . . . . .	110
6.31.4.3	clear . . . . .	111
6.31.4.4	getAllowedValueList . . . . .	111
6.31.4.5	getAttribute . . . . .	111
6.31.4.6	getDataType . . . . .	111
6.31.4.7	getValue . . . . .	111
6.31.4.8	operator= . . . . .	111
6.31.4.9	sendEvents . . . . .	111
6.31.4.10	setAttribute . . . . .	111
<b>7</b>	<b>File Documentation</b> . . . . .	<b>113</b>
7.1	src/core/brisacconfig.cpp File Reference . . . . .	113
7.2	src/core/brisacconfig.h File Reference . . . . .	114
7.3	src/core/brisacore.h File Reference . . . . .	115
7.4	src/core/brisaglobal.h File Reference . . . . .	116
7.4.1	Define Documentation . . . . .	116
7.4.1.1	BRISA_CORE_EXPORT . . . . .	116
7.4.1.2	BRISA_UPNP_EXPORT . . . . .	116
7.4.1.3	BRISA_UTILS_EXPORT . . . . .	116
7.4.1.4	BRISA_VERSION . . . . .	116
7.4.1.5	BRISA_VERSION_STR . . . . .	116
7.5	src/core/brisawebserver.cpp File Reference . . . . .	117
7.5.1	Function Documentation . . . . .	117
7.5.1.1	extractPathLevel . . . . .	117

7.6	src/core/brisawebserver.h File Reference . . . . .	118
7.6.1	Define Documentation . . . . .	119
7.6.1.1	DEFAULT_PAGE . . . . .	119
7.7	src/upnp/brisabSTRACTeventsSubscription.cpp File Reference . . . . .	120
7.8	src/upnp/brisabSTRACTeventsSubscription.h File Reference . . . . .	121
7.9	src/upnp/brisabSTRACTservice.cpp File Reference . . . . .	122
7.10	src/upnp/brisabSTRACTservice.h File Reference . . . . .	123
7.11	src/upnp/brisaction.cpp File Reference . . . . .	124
7.12	src/upnp/brisaction.h File Reference . . . . .	125
7.13	src/upnp/brisargument.cpp File Reference . . . . .	126
7.14	src/upnp/brisargument.h File Reference . . . . .	127
7.15	src/upnp/brisaticon.cpp File Reference . . . . .	128
7.16	src/upnp/brisaticon.h File Reference . . . . .	129
7.17	src/upnp/brisaservicexmlHandler.cpp File Reference . . . . .	130
7.18	src/upnp/brisaservicexmlHandler.h File Reference . . . . .	131
7.19	src/upnp/brisastatevariable.cpp File Reference . . . . .	133
7.20	src/upnp/brisastatevariable.h File Reference . . . . .	134
7.21	src/upnp/controlpoint/brisacontrolpoint.cpp File Reference . . . . .	135
7.22	src/upnp/controlpoint/brisacontrolpoint.h File Reference . . . . .	136
7.23	src/upnp/controlpoint/brisacontrolpointdevice.cpp File Reference . . . . .	137
7.24	src/upnp/controlpoint/brisacontrolpointdevice.h File Reference . . . . .	138
7.25	src/upnp/controlpoint/brisacontrolpointservice.cpp File Reference . . . . .	139
7.26	src/upnp/controlpoint/brisacontrolpointservice.h File Reference . . . . .	140
7.27	src/upnp/controlpoint/brisadevicexmlHandlercp.cpp File Reference . . . . .	141
7.28	src/upnp/controlpoint/brisadevicexmlHandlercp.h File Reference . . . . .	142
7.29	src/upnp/controlpoint/brisaeventproxy.cpp File Reference . . . . .	144
7.30	src/upnp/controlpoint/brisaeventproxy.h File Reference . . . . .	145
7.31	src/upnp/controlpoint/brisamsearchclientcp.cpp File Reference . . . . .	146
7.31.1	Define Documentation . . . . .	146
7.31.1.1	UPNP_MSEARCH_DISCOVER . . . . .	146
7.32	src/upnp/controlpoint/brisamsearchclientcp.h File Reference . . . . .	147
7.32.1	Define Documentation . . . . .	147
7.32.1.1	DEFAULT_SEARCH_TIME . . . . .	147
7.32.1.2	DEFAULT_SEARCH_TYPE . . . . .	147
7.33	src/upnp/device/brisactionxmlparser.cpp File Reference . . . . .	148
7.34	src/upnp/device/brisactionxmlparser.h File Reference . . . . .	149

7.35 src/upnp/device/brisadevice.cpp File Reference . . . . .	150
7.36 src/upnp/device/brisadevice.h File Reference . . . . .	151
7.37 src/upnp/device/brisadevicexmlhandler.cpp File Reference . . . . .	152
7.38 src/upnp/device/brisadevicexmlhandler.h File Reference . . . . .	153
7.39 src/upnp/device/brisaeventcontroller.cpp File Reference . . . . .	154
7.39.1 Define Documentation . . . . .	154
7.39.1.1 ERROR_400_MESSAGE . . . . .	154
7.39.1.2 ERROR_412_MESSAGE . . . . .	154
7.40 src/upnp/device/brisaeventcontroller.h File Reference . . . . .	155
7.41 src/upnp/device/brisaeventmessage.cpp File Reference . . . . .	156
7.42 src/upnp/device/brisaeventmessage.h File Reference . . . . .	157
7.43 src/upnp/device/brisaeventssubscription.cpp File Reference . . . . .	158
7.44 src/upnp/device/brisaeventssubscription.h File Reference . . . . .	159
7.45 src/upnp/device/brisaservice.cpp File Reference . . . . .	160
7.45.1 Define Documentation . . . . .	160
7.45.1.1 SOAP_ERROR_TEMPLATE . . . . .	160
7.46 src/upnp/device/brisaservice.h File Reference . . . . .	161
7.47 src/upnp/ssdp/brisassdclient.cpp File Reference . . . . .	162
7.48 src/upnp/ssdp/brisassdclient.h File Reference . . . . .	163
7.49 src/upnp/ssdp/brisassdpserver.cpp File Reference . . . . .	164
7.49.1 Define Documentation . . . . .	164
7.49.1.1 UPNP_ALIVE_MESSAGE . . . . .	164
7.49.1.2 UPNP_BYEBYE_MESSAGE . . . . .	164
7.49.1.3 UPNP_MSEARCH_RESPONSE . . . . .	165
7.50 src/upnp/ssdp/brisassdpserver.h File Reference . . . . .	166
7.51 src/utils/brisalog.cpp File Reference . . . . .	167
7.51.1 Define Documentation . . . . .	167
7.51.1.1 COLOR_CRITICAL . . . . .	167
7.51.1.2 COLOR_DEBUG . . . . .	167
7.51.1.3 COLOR_FATAL . . . . .	167
7.51.1.4 COLOR_RESET . . . . .	168
7.51.1.5 COLOR_WARN . . . . .	168
7.51.1.6 LOG_WRITE . . . . .	168
7.51.2 Function Documentation . . . . .	168
7.51.2.1 brisaLogInitialize . . . . .	168
7.51.2.2 brisaMessageWritter . . . . .	168

<b>7.52 src/utils/brisalog.h File Reference</b>	169
<b>7.52.1 Function Documentation</b>	169
<b>7.52.1.1 brisaLogInitialize</b>	169
<b>7.53 src/utils/brisagraph.cpp File Reference</b>	170
<b>7.53.1 Function Documentation</b>	170
<b>7.53.1.1 getIp</b>	170
<b>7.53.1.2 getPort</b>	170
<b>7.54 src/utils/brisagraph.h File Reference</b>	171
<b>7.54.1 Function Documentation</b>	171
<b>7.54.1.1 getIp</b>	171
<b>7.54.1.2 getPort</b>	171



# Chapter 1

## Namespace Index

### 1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

BrisaCore . . . . .	9
BrisaUpnp . . . . .	10



# Chapter 2

## Class Index

### 2.1 Class Hierarchy

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

BrisaCore::BrisaConfigurationManager . . . . .	15
BrisaCore::BrisaWebFile . . . . .	19
BrisaCore::BrisaWebserver . . . . .	20
BrisaCore::BrisaWebService . . . . .	23
BrisaUpnp::BrisaEventController . . . . .	75
BrisaCore::BrisaWebServiceProvider . . . . .	26
BrisaCore::BrisaWebStaticContent . . . . .	29
BrisaUpnp::BrisaAbstractEventSubscription . . . . .	30
BrisaUpnp::BrisaEventProxy . . . . .	81
BrisaUpnp::BrisaEventSubscription . . . . .	85
BrisaUpnp::BrisaAbstractService . . . . .	33
BrisaUpnp::BrisaControlPointService . . . . .	55
BrisaUpnp::BrisaService . . . . .	91
BrisaUpnp::BrisaAction . . . . .	38
BrisaUpnp::BrisaActionXmlParser . . . . .	42
BrisaUpnp::BrisaArgument . . . . .	44
BrisaUpnp::BrisaControlPoint . . . . .	45
BrisaUpnp::BrisaControlPointDevice . . . . .	50
BrisaUpnp::BrisaDevice . . . . .	58
BrisaUpnp::BrisaDeviceParserContext . . . . .	67
BrisaUpnp::BrisaDeviceXMLHandler . . . . .	70
BrisaUpnp::BrisaDeviceXMLHandlerCP . . . . .	71
BrisaUpnp::BrisaEventMessage . . . . .	79
BrisaUpnp::BrisaIcon . . . . .	88
BrisaUpnp::BrisaMSearchClientCP . . . . .	89
BrisaUpnp::BrisaServiceFetcher . . . . .	95
BrisaUpnp::BrisaServiceParserContext . . . . .	98
BrisaUpnp::BrisaServiceXMLHandler . . . . .	101
BrisaUpnp::BrisaSSDPClient . . . . .	104
BrisaUpnp::BrisaSSDPServer . . . . .	106
BrisaUpnp::BrisaStateVariable . . . . .	109



# Chapter 3

## Class Index

### 3.1 Class List

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

BrisaCore::BrisaConfigurationManager (Class that provides an easy way of managing configurations ) . . . . .	15
BrisaCore::BrisaWebFile (Adds a file to the web server ) . . . . .	19
BrisaCore::BrisaWebserver (The BrisaWebserver class is a web server implementation ) . . . . .	20
BrisaCore::BrisaWebService (Web service abstraction class ) . . . . .	23
BrisaCore::BrisaWebServiceProvider (The BrisaWebServiceProvider class works as web service manager for the web server ) . . . . .	26
BrisaCore::BrisaWebStaticContent (The BrisaWebStaticContent class stores a QString into the web server ) . . . . .	29
BrisaUpnp::BrisaAbstractEventSubscription . . . . .	30
BrisaUpnp::BrisaAbstractService . . . . .	33
BrisaUpnp::BrisaAction (Template method class that represents each service's action ) . . . . .	38
BrisaUpnp::BrisaActionXmlParser (XML parser for SOAP requests ) . . . . .	42
BrisaUpnp::BrisaArgument . . . . .	44
BrisaUpnp::BrisaControlPoint (Class that implements the control part in UPnP Architecture ) . . . . .	45
BrisaUpnp::BrisaControlPointDevice (Class that implements the devices that control point part is going to handle ) . . . . .	50
BrisaUpnp::BrisaControlPointService (BrisaControlPointService is the class that implements action control in UPnP Architecture ) . . . . .	55
BrisaUpnp::BrisaDevice (UPnP device implementation ) . . . . .	58
BrisaUpnp::BrisaDeviceParserContext . . . . .	67
BrisaUpnp::BrisaDeviceXMLHandler . . . . .	70
BrisaUpnp::BrisaDeviceXMLHandlerCP (BrisaDeviceXMLHandlerCP creates a device from a xml description file, with all it's attributes, it let's it ready to be used ) . . . . .	71
BrisaUpnp::BrisaEventController . . . . .	75
BrisaUpnp::BrisaEventMessage . . . . .	79
BrisaUpnp::BrisaEventProxy (Class that implements the event part in control point side in Brisa Qt, this class makes the operations of subscribe, renew subscription and unsubscribe ) . . . . .	81
BrisaUpnp::BrisaEventSubscription . . . . .	85
BrisaUpnp::BrisaIcon . . . . .	88
BrisaUpnp::BrisaMSearchClientCP (SSDP MSearch implementation for UPnP control points ) . . . . .	89
BrisaUpnp::BrisaService (UPnP service abstraction ) . . . . .	91
BrisaUpnp::BrisaServiceFetcher . . . . .	95

<a href="#">BrisaUpnp::BrisaServiceParserContext</a>	98
<a href="#">BrisaUpnp::BrisaServiceXMLHandler</a>	101
<a href="#">BrisaUpnp::BrisaSSDPClient</a> (SSDP stack implementantion for UPnP control points )	104
<a href="#">BrisaUpnp::BrisaSSDPServer</a> (SSDP stack implementation for UPnP devices )	106
<a href="#">BrisaUpnp::BrisaStateVariable</a> (Represents the service's state variables )	109

# Chapter 4

## File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

src/core/ <a href="#">brisacconfig.cpp</a>	113
src/core/ <a href="#">brisacconfig.h</a>	114
src/core/ <a href="#">brisacore.h</a>	115
src/core/ <a href="#">brisaglobal.h</a>	116
src/core/ <a href="#">brisawebserver.cpp</a>	117
src/core/ <a href="#">brisawebserver.h</a>	118
src/upnp/ <a href="#">brisaabstacteventsbscription.cpp</a>	120
src/upnp/ <a href="#">brisaabstacteventsbscription.h</a>	121
src/upnp/ <a href="#">brisaabstactservice.cpp</a>	122
src/upnp/ <a href="#">brisaabstactservice.h</a>	123
src/upnp/ <a href="#">brisaaaction.cpp</a>	124
src/upnp/ <a href="#">brisaaaction.h</a>	125
src/upnp/ <a href="#">brisaaargument.cpp</a>	126
src/upnp/ <a href="#">brisaaargument.h</a>	127
src/upnp/ <a href="#">brisaiicon.cpp</a>	128
src/upnp/ <a href="#">brisaiicon.h</a>	129
src/upnp/ <a href="#">brisaservicexmlhandler.cpp</a>	130
src/upnp/ <a href="#">brisaservicexmlhandler.h</a>	131
src/upnp/ <a href="#">brisastatevariable.cpp</a>	133
src/upnp/ <a href="#">brisastatevariable.h</a>	134
src/upnp/controlpoint/ <a href="#">brisaccontrolpoint.cpp</a>	135
src/upnp/controlpoint/ <a href="#">brisaccontrolpoint.h</a>	136
src/upnp/controlpoint/ <a href="#">brisaccontrolpointdevice.cpp</a>	137
src/upnp/controlpoint/ <a href="#">brisaccontrolpointdevice.h</a>	138
src/upnp/controlpoint/ <a href="#">brisaccontrolpointservice.cpp</a>	139
src/upnp/controlpoint/ <a href="#">brisaccontrolpointservice.h</a>	140
src/upnp/controlpoint/ <a href="#">brisadevicexmlhandlercp.cpp</a>	141
src/upnp/controlpoint/ <a href="#">brisadevicexmlhandlercp.h</a>	142
src/upnp/controlpoint/ <a href="#">brisaeventproxy.cpp</a>	144
src/upnp/controlpoint/ <a href="#">brisaeventproxy.h</a>	145
src/upnp/controlpoint/ <a href="#">brisamsearchclientcp.cpp</a>	146
src/upnp/controlpoint/ <a href="#">brisamsearchclientcp.h</a>	147
src/upnp/device/ <a href="#">brisaactionxmlparser.cpp</a>	148

src/upnp/device/ <a href="#">brisactionxmlparser.h</a>	149
src/upnp/device/ <a href="#">brisadvice.cpp</a>	150
src/upnp/device/ <a href="#">brisadvice.h</a>	151
src/upnp/device/ <a href="#">brisadvicexmlhandler.cpp</a>	152
src/upnp/device/ <a href="#">brisadvicexmlhandler.h</a>	153
src/upnp/device/ <a href="#">brisaeventcontroller.cpp</a>	154
src/upnp/device/ <a href="#">brisaeventcontroller.h</a>	155
src/upnp/device/ <a href="#">brisaeventmessage.cpp</a>	156
src/upnp/device/ <a href="#">brisaeventmessage.h</a>	157
src/upnp/device/ <a href="#">brisaeventsubscription.cpp</a>	158
src/upnp/device/ <a href="#">brisaeventsubscription.h</a>	159
src/upnp/device/ <a href="#">brisaservice.cpp</a>	160
src/upnp/device/ <a href="#">brisaservice.h</a>	161
src/upnp/ssdp/ <a href="#">brisassdpclient.cpp</a>	162
src/upnp/ssdp/ <a href="#">brisassdpclient.h</a>	163
src/upnp/ssdp/ <a href="#">brisassdpserver.cpp</a>	164
src/upnp/ssdp/ <a href="#">brisassdpserver.h</a>	166
src/utils/ <a href="#">brisalog.cpp</a>	167
src/utils/ <a href="#">brisalog.h</a>	169
src/utils/ <a href="#">brisanetwork.cpp</a>	170
src/utils/ <a href="#">brisanetwork.h</a>	171

# Chapter 5

## Namespace Documentation

### 5.1 BrisaCore Namespace Reference

#### Classes

- class [BrisaConfigurationManager](#)

*Class that provides an easy way of managing configurations.*

- class [BrisaWebService](#)

*Web service abstraction class.*

- class [BrisaWebFile](#)

*Adds a file to the web server.*

- class [BrisaWebStaticContent](#)

*The [BrisaWebStaticContent](#) class stores a `QString` into the web server.*

- class [BrisaWebServiceProvider](#)

*The [BrisaWebServiceProvider](#) class works as web service manager for the web server.*

- class [BrisaWebserver](#)

*The [BrisaWebserver](#) class is a web server implementation.*

## 5.2 BrisaUpnp Namespace Reference

### Classes

- class [BrisaAbstractEventSubscription](#)
- class [BrisaAbstractService](#)
- class [BrisaAction](#)

*Template method class that represents each service's action.*

- class [BrisaArgument](#)
- class [BrisaIcon](#)
- class [BrisaServiceParserContext](#)
- class [BrisaServiceXMLHandler](#)
- class [BrisaStateVariable](#)

*Represents the service's state variables.*

- class [BrisaControlPoint](#)

*Class that implements the control part in UPnP Architecture.*

- class [BrisaControlPointDevice](#)

*Class that implements the devices that control point part is going to handle.*

- class [BrisaControlPointService](#)

*BrisaControlPointService is the class that implements action control in UPnP Architecture.*

- class [BrisaDeviceParserContext](#)
- class [BrisaDeviceXMLHandlerCP](#)

*BrisaDeviceXMLHandlerCP creates a device from a xml description file, with all it's attributes, it let's it ready to be used.*

- class [BrisaServiceFetcher](#)
- class [BrisaEventProxy](#)

*Class that implements the event part in control point side in Brisa Qt, this class makes the operations of subscribe, renew subscription and unsubscribe.*

- class [BrisaMSearchClientCP](#)

*SSDP MSearch implementation for UPnP control points.*

- class [BrisaActionXmlParser](#)

*XML parser for SOAP requests.*

- class [BrisaDevice](#)

*UPnP device implementation.*

- class [BrisaDeviceXMLHandler](#)
- class [BrisaEventController](#)
- class [BrisaEventMessage](#)
- class [BrisaEventSubscription](#)
- class [BrisaService](#)

*UPnP service abstraction.*

- class [BrisaSSDPClient](#)  
*SSDP stack implementantion for UPnP control points.*
- class [BrisaSSDPServer](#)  
*SSDP stack implementation for UPnP devices.*

## Enumerations

- enum [saxParserState](#) {
 ServiceStart, Scpd, ServiceSpecVersion, ServiceSpecVersionMajor,
 ServiceSpecVersionMinor, ActionList, Action, ActionName,
 ArgumentList, Argument, ArgumentName, ArgumentDirection,
 RelatedStateVariable, ServiceStateTable, StateVariable, StateVariableName,
 StateVariableDataType, StateVariableDefaultValue, StateVariableAllowedValueList, StateVariableAllowedValue,
 StateVariableAllowedValueRange, StateVariableAllowedValueRangeMinimum, StateVariableAllowedValueRangeMaximum, StateVariableAllowedValueRangeStep,
 ServiceFinished, ServiceError = -1 }
- enum [SaxParserState](#) {
 Start, Root, SpecVersion, SpecVersionMajor,
 SpecVersionMinor, UrlBase, Device, DeviceType,
 DeviceFriendlyName, Manufacturer, ManufacturerUrl, ModelDescription,
 ModelName, ModelUrl, SerialNumber, Udn,
 Upc, IconList, Icon, IconMimetype,
 IconWidth, IconHeight, IconDepth, IconUrl,
 PresentationUrl, DeviceList, ServiceList, Service,
 ServiceType, ServiceId, ServiceScpdUrl, ServiceControlUrl,
 ServiceEventSubUrl, Finished, Error = -1 }

### 5.2.1 Enumeration Type Documentation

#### 5.2.1.1 enum BrisaUpnp::SaxParserState

**Enumerator:**

*Start*  
*Root*  
*SpecVersion*  
*SpecVersionMajor*  
*SpecVersionMinor*  
*UrlBase*  
*Device*  
*DeviceType*

*DeviceFriendlyName*  
*Manufacturer*  
*ManufacturerUrl*  
*ModelDescription*  
*ModelName*  
*ModelUrl*  
*SerialNumber*  
*Udn*  
*Upc*  
*IconList*  
*Icon*  
*IconMimetype*  
*IconWidth*  
*IconHeight*  
*IconDepth*  
*IconUrl*  
*PresentationUrl*  
*DeviceList*  
*ServiceList*  
*Service*  
*ServiceType*  
*ServiceId*  
*ServiceScpdUrl*  
*ServiceControlUrl*  
*ServiceEventSubUrl*  
*Finished*  
*Error*

### 5.2.1.2 enum BrisaUpnp::saxParserState

Enumerator:

*ServiceStart*  
*Scpd*  
*ServiceSpecVersion*  
*ServiceSpecVersionMajor*  
*ServiceSpecVersionMinor*  
*ActionList*  
*Action*  
*ActionName*  
*ArgumentList*  
*Argument*

*ArgumentName*  
*ArgumentDirection*  
*RelatedStateVariable*  
*ServiceStateTable*  
*State Variable*  
*StateVariableName*  
*StateVariableDataType*  
*StateVariableDefaultValue*  
*StateVariableAllowedValueList*  
*StateVariableAllowedValue*  
*StateVariableAllowedValueRange*  
*StateVariableAllowedValueRangeMinimum*  
*StateVariableAllowedValueRangeMaximum*  
*StateVariableAllowedValueRangeStep*  
*ServiceFinished*  
*ServiceError*



# Chapter 6

## Class Documentation

### 6.1 BrisaCore::BrisaConfigurationManager Class Reference

Class that provides an easy way of managing configurations.

```
#include <BrisaCore/BrisaConfigurationManager>
```

#### Public Member Functions

- `BrisaConfigurationManager` (const QString &configPath, const QHash< QString, QString > &state)  
*Constructor for the `BrisaConfigurationManager` class.*
- void `setDirectAccess` (bool access)  
*Sets the direct access option of the ConfigurationManager.*
- bool `getDirectAccess` ()  
*Returns False if the ConfigurationManager is currently working on the runtime state.*
- void `update` ()  
*Updates the current state of the manager according to persistence data.*
- void `save` ()  
*Stores the state of the manager on the persistence.*
- QString `getParameter` (const QString &section, const QString &parameter)  
*Retrieves the value associated with the parameter in the section given.*
- void `setParameter` (const QString &section, const QString &parameter, const QString &parValue)  
*Sets a parameter's value in the given section.*
- QList< QString > `getSectionNames` ()  
*Returns the names of all sections.*
- QHash< QString, QString > `items` (const QString &section)

*Returns all the items of the given section.*

- bool [removeSection](#) (const QString &section)

*Removes a section given the name.*

### 6.1.1 Detailed Description

Class that provides an easy way of managing configurations. Configuration is organized in sections, which may contain parameters. Each section has a name and a parameter has a name and a value.

Concerning storage, the configuration can be saved on a sqlite database. Also, there's a feature called "direct access" that enables direct operations on the database. For example,a [getParameter\(\)](#) call would retrieve the value of a parameter directly from the database, if the feature is enabled.

When disabled, all methods apply to (what we call) the "state". The state initially contains the same information that is on the database, but if the "direct access" feature is disabled, all get()'s and set()'s apply to the state. This means you can have a "runtime configuration" and a "static configuration".

By default, the "direct access" feature is disabled. To enable it, just call [setDirectAccess\(True\)](#).

The state can be saved on the persistence by explicitly calling [save\(\)](#) It can also update its values by explicitly calling [update\(\)](#).

### 6.1.2 Constructor & Destructor Documentation

#### 6.1.2.1 BrisaConfigurationManager::BrisaConfigurationManager (const QString & configPath, const QHash<QString, QString > & state)

Constructor for the [BrisaConfigurationManager](#) class.

##### Parameters:

*config\_path* a QString that represents the path of the database to work on. If not supplied will work on a memory database.

*state*,: hash as a dictionaryr with sections and parameters.\ Keys are section.parameters format and values are the their respective values.

### 6.1.3 Member Function Documentation

#### 6.1.3.1 bool BrisaConfigurationManager::getDirectAccess ()

Returns False if the ConfigurationManager is currently working on the runtime state. Otherwise, it will return True, which means it's working directly on the persistence.

##### Returns:

The current status of the ConfigurationManager

Referenced by [getParameter\(\)](#), [getSectionNames\(\)](#), [items\(\)](#), [removeSection\(\)](#), and [setParameter\(\)](#).

**6.1.3.2 QString BrisaConfigurationManager::getParameter (const QString & *section*, const QString & *parameter*)**

Retrieves the value associated with the parameter in the section given.

**Parameters:**

*section* a section to find the parameter

*parameter* a parameter to return the value

**Returns:**

the value for the given parameter

**6.1.3.3 QList<QString> BrisaConfigurationManager::getSectionNames ()**

Returns the names of all sections.

**Returns:**

a QList with its parameters and values

**6.1.3.4 QHash<QString, QString> BrisaConfigurationManager::items (const QString & *section*)**

Returns all the items of the given section.

**Parameters:**

*section* name

**Returns:**

a dictionary with all the items of the given section

Referenced by removeSection().

**6.1.3.5 bool BrisaConfigurationManager::removeSection (const QString & *section*)**

Removes a section given the name.

**Parameters:**

*section*,: section name to be removed

**Returns:**

a boolean. true whether is possible remove it

**6.1.3.6 void BrisaConfigurationManager::save ()**

Stores the state of the manager on the persistence.

Referenced by removeSection(), and setParameter().

### 6.1.3.7 void BrisaConfigurationManager::setDirectAccess (bool *access*)

Sets the direct access option of the ConfigurationManager. When True, direct access makes all get and set methods work directly on the database, not on the current state.

**Parameters:**

*access* a boolean argument to set direct access option of the [BrisaConfigurationManager](#)

### 6.1.3.8 void BrisaConfigurationManager::setParameter (const QString & *section*, const QString & *parameter*, const QString & *parValue*)

Sets a parameter's value in the given section. If the parameter does not exist, it gets created.

**Parameters:**

*section* a section to set the parameter

*parameter* a parameter to set the value

*parValue* a value to be set

### 6.1.3.9 void BrisaConfigurationManager::update ()

Updates the current state of the manager according to persistence data.

Referenced by `getParameter()`, `getSectionNames()`, and `items()`.

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

- `src/core/brisacconfig.h`
- `src/core/brisacconfig.cpp`

## 6.2 BrisaCore::BrisaWebFile Class Reference

Adds a file to the web server.

```
#include <BrisaCore/BrisaWebFile>
```

### Public Member Functions

- [BrisaWebFile \(QxtAbstractWebSessionManager \\*sm, QString filePath, QObject \\*parent=0\)](#)  
*Constructor for BrisaWebFile.*
- [~BrisaWebFile \(\)](#)  
*Destructor for BrisaWebFile.*
- [void pageRequestedEvent \(QxtWebRequestEvent \\*event\)](#)  
*Reimplemented from libQxt.*

### 6.2.1 Detailed Description

Adds a file to the web server. Use this class to store a file into the web server. If the [BrisaWebFile](#) is stored using a [BrisaWebServiceProvider](#), it's url path will be of "IP:PORT/SERVICENAME/yourfile". if it's stored using the BrisaWebServer convenience method "*publishFile()*", it's url path will be "IP:PORT/yourfile".

### 6.2.2 Constructor & Destructor Documentation

#### 6.2.2.1 BrisaWebFile::BrisaWebFile (QxtAbstractWebSessionManager \* *sm*, QString *filePath*, QObject \* *parent* = 0) [inline]

Constructor for [BrisaWebFile](#). It creates a QFile with the given file path.

#### 6.2.2.2 BrisaWebFile::~BrisaWebFile () [inline]

Destructor for [BrisaWebFile](#).

### 6.2.3 Member Function Documentation

#### 6.2.3.1 void BrisaWebFile::pageRequestedEvent (QxtWebRequestEvent \* *event*) [inline]

Reimplemented from libQxt. When a request is received the [BrisaWebFile](#) will reply the stored file.

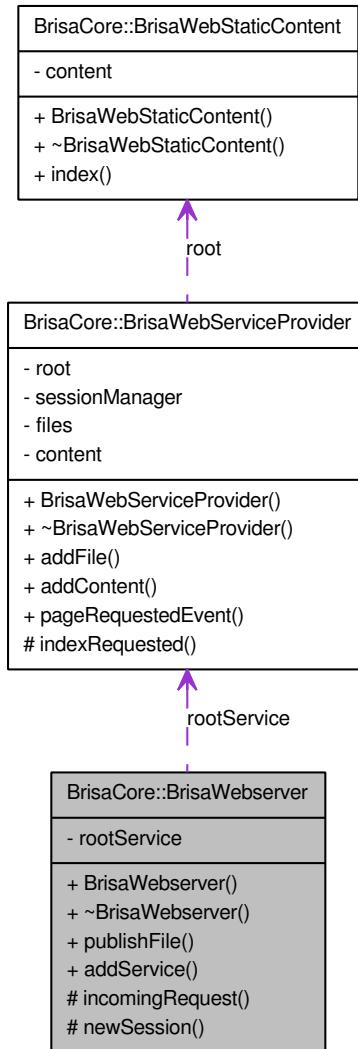
The documentation for this class was generated from the following files:

- [src/core/brisawebserver.h](#)
- [src/core/brisawebserver.cpp](#)

## 6.3 BrisaCore::BrisaWebserver Class Reference

The [BrisaWebserver](#) class is a web server implementation.

#include <BrisaCore/BrisaWebserver>Collaboration diagram for BrisaCore::BrisaWebserver:



### Public Member Functions

- [BrisaWebserver](#) (const QHostAddress &host, quint16 port)
 

*Constructor for BrisaWebServer.*
- [~BrisaWebserver](#) ()
 

*Destructor for BrisaWebServer.*
- void [publishFile](#) (QString publishPath, QString filePath)
 

*Publishes a file to the root.*

- void [addService](#) (QString path, QxtWebServiceDirectory \*service)  
*Adds a service to the web server.*

## Protected Member Functions

- void [incomingRequest](#) (quint32 requestID, const QHttpRequestHeader &header, QxtWebContent \*deviceContent)
- int [newSession](#) ()  
*Creates a new session and returns the session number.*

### 6.3.1 Detailed Description

The [BrisaWebserver](#) class is a web server implementation. BrisaWebServer implements a Web Server using libQxt.

### 6.3.2 Constructor & Destructor Documentation

#### 6.3.2.1 BrisaWebserver::BrisaWebserver (const QHostAddress & host, quint16 port)

Constructor for BrisaWebServer.

#### 6.3.2.2 BrisaWebserver::~BrisaWebserver ()

Destructor for BrisaWebServer.

### 6.3.3 Member Function Documentation

#### 6.3.3.1 void BrisaWebserver::addService (QString path, QxtWebServiceDirectory \* service)

Adds a service to the web server. The service url path will be added to the root of the server.

#### 6.3.3.2 void BrisaWebserver::incomingRequest (quint32 requestID, const QHttpRequestHeader & header, QxtWebContent \* deviceContent) [protected]

#### 6.3.3.3 int BrisaWebserver::newSession () [protected]

Creates a new session and returns the session number.

#### 6.3.3.4 void BrisaWebserver::publishFile (QString publishPath, QString filePath)

Publishes a file to the root.

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

- src/core/[brisawebserver.h](#)

- src/core/[brisawebserver.cpp](#)

## 6.4 BrisaCore::BrisaWebService Class Reference

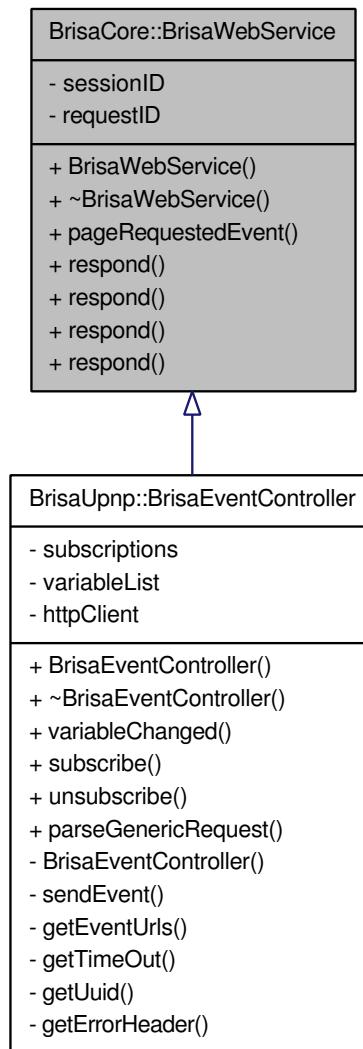
Web service abstraction class.

```
#include <BrisaCore/BrisaWebService>Inheritance
```

diagram

for

BrisaCore::BrisaWebService:



### Public Slots

- void [pageRequestedEvent](#) (QxtWebRequestEvent \*event)

*Reimplemented from libQxt.*

- void [respond](#) (QByteArray response)

*Responds response to the session and request ID currently stored in [BrisaWebService](#), if using this method the response must be synchronous because the request and session ID can change quickly.*

- void [respond](#) (const QByteArray &response, const int &sessionId, const int &requestId)

*Reimplements [respond\(\)](#).*

- void [respond](#) (const QHttpResponseHeader &response)

*Reimplements [respond\(\)](#) This method responds only a HTTP header to the session and request ID stored in BrisaWebService.*

- void [respond](#) (const QHttpResponseHeader &response, const int &sessionId, const int &requestId)

*Reimplements [respond\(\)](#).*

## Signals

- void [genericRequestReceived](#) (const QString &method, const QMultiHash< QString, QString > &headers, const QByteArray &requestContent, int sessionId, int requestId)

*This signal is emitted when BrisaWebService receives a request.*

- void [genericRequestReceived](#) (BrisaWebService \*service, QMultiHash< QString, QString >, QString requestContent)

*Reimplements [genericRequestReceived\(\)](#) This signal is emitted when BrisaWebService receives a request, the main difference is that this signal has a pointer to the class that is emitting the signal.*

## Public Member Functions

- [BrisaWebService](#) (QxtAbstractWebSessionManager \*sm, QObject \*parent=0)

*Constructor for BrisaWebService.*

- [~BrisaWebService](#) ()

*Destructor for BrisaWebService.*

### 6.4.1 Detailed Description

Web service abstraction class. [BrisaWebService](#) is used to receive and respond UPnP action and event requests. Currently this class is used mostly with BrisaService and BrisaEventController.

#### See also:

[BrisaUpnp::BrisaService](#) , [BrisaUpnp::BrisaEventController](#)

### 6.4.2 Constructor & Destructor Documentation

#### 6.4.2.1 [BrisaWebService::BrisaWebService](#) (QxtAbstractWebSessionManager \* *sm*, QObject \* *parent* = 0) [inline]

Constructor for [BrisaWebService](#).

#### 6.4.2.2 [BrisaWebService::~BrisaWebService](#) () [inline]

Destructor for [BrisaWebService](#).

### 6.4.3 Member Function Documentation

**6.4.3.1 void BrisaWebService::genericRequestReceived (BrisaWebService \* *service*, QMultiHash< QString, QString >, QString *requestContent*) [signal]**

Reimplements [genericRequestReceived\(\)](#) This signal is emmited when [BrisaWebService](#) receives a request, the main difference is that this signal has a pointer to the class that is emmiting the signal.

**6.4.3.2 void BrisaWebService::genericRequestReceived (const QString & *method*, const QMultiHash< QString, QString > & *headers*, const QByteArray & *requestContent*, int *sessionId*, int *requestId*) [signal]**

This signal is emmited when [BrisaWebService](#) receives a request.

Referenced by [BrisaUpnp::BrisaEventController::BrisaEventController\(\)](#).

**6.4.3.3 void BrisaWebService::pageRequestedEvent (QxtWebRequestEvent \* *event*) [inline, slot]**

Reimplemented from libQxt. This method receives all web service requests and emits a [genericRequestReceived\(\)](#) signal. If the request method is of "POST" type, the web service will reply a default message.

**6.4.3.4 void BrisaWebService::respond (const QHttpResponseHeader & *response*, const int & *sessionId*, const int & *requestId*) [inline, slot]**

Reimplements [respond\(\)](#). This method responds only a HTTP header using the given session and request ID.

**6.4.3.5 void BrisaWebService::respond (const QHttpResponseHeader & *response*) [inline, slot]**

Reimplements [respond\(\)](#) This method responds only a HTTP header to the session and request ID stored in [BrisaWebService](#).

**6.4.3.6 void BrisaWebService::respond (const QByteArray & *response*, const int & *sessionId*, const int & *requestId*) [inline, slot]**

Reimplements [respond\(\)](#). We recommend using this method given the fact that it supports asynchronous requests.

**6.4.3.7 void BrisaWebService::respond (QByteArray *response*) [inline, slot]**

Responds *response* to the session and request ID currently stored in [BrisaWebService](#), if using this method the response must be synchronous because the request and session ID can change quickly.

Referenced by [BrisaUpnp::BrisaEventController::subscribe\(\)](#), and [BrisaUpnp::BrisaEventController::unsubscribe\(\)](#).

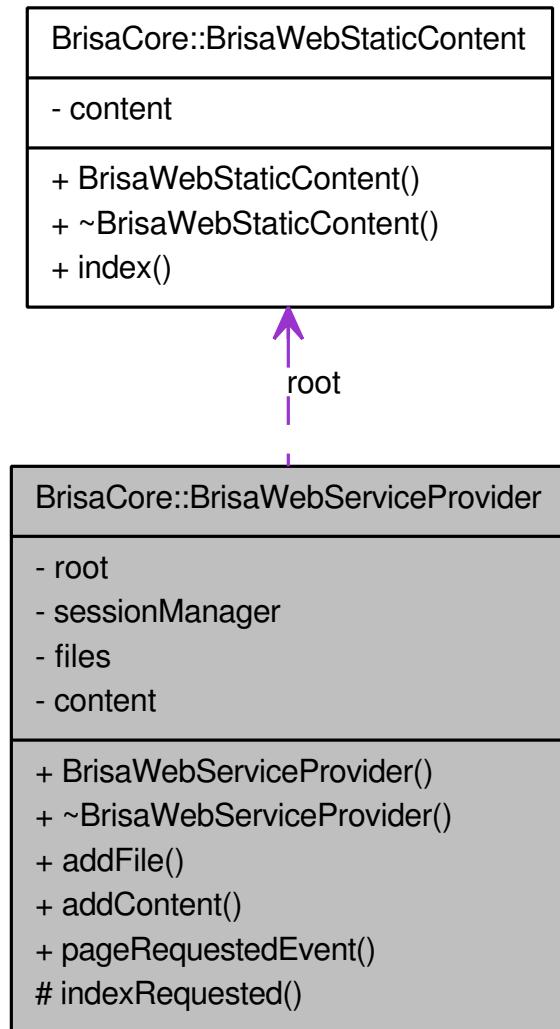
The documentation for this class was generated from the following files:

- [src/core/brisawebserver.h](#)
- [src/core/brisawebserver.cpp](#)

## 6.5 BrisaCore::BrisaWebServiceProvider Class Reference

The [BrisaWebServiceProvider](#) class works as web service manager for the web server.

#include <BrisaCore/BrisaWebServiceProvider>Collaboration diagram for BrisaCore::BrisaWebServiceProvider:



### Public Member Functions

- `BrisaWebServiceProvider` (QxtAbstractWebSessionManager \*sm, QObject \*parent)  
*Constructor for `BrisaWebServiceProvider`.*
- `~BrisaWebServiceProvider ()`  
*Destructor for `BrisaWebServiceProvider`.*
- void `addFile` (const QString path, QString filePath)  
*Call this method to add a `BrisaWebFile` to the web service.*

- void [addContent](#) (const QString path, QString content)  
*Call this method to add a [BrisaWebStaticContent](#) to the web service.*
- void [pageRequestedEvent](#) (QxtWebRequestEvent \*event)

## Protected Member Functions

- void [indexRequested](#) (QxtWebRequestEvent \*event)

### 6.5.1 Detailed Description

The [BrisaWebServiceProvider](#) class works as web service manager for the web server. The [BrisaWebServiceProvider](#) has convenience methods for managing web services, like [addFile\(\)](#) and [addContent\(\)](#). It also keeps track of all files and content stored into the web service.

### 6.5.2 Constructor & Destructor Documentation

#### 6.5.2.1 [BrisaWebServiceProvider::BrisaWebServiceProvider](#) ([QxtAbstractWebSessionManager](#) \* *sm*, [QObject](#) \* *parent*) [inline]

Constructor for [BrisaWebServiceProvider](#).

#### 6.5.2.2 [BrisaWebServiceProvider::~BrisaWebServiceProvider](#) () [inline]

Destructor for [BrisaWebServiceProvider](#).

### 6.5.3 Member Function Documentation

#### 6.5.3.1 [void BrisaWebServiceProvider::addContent](#) (const QString *path*, QString *content*)

Call this method to add a [BrisaWebStaticContent](#) to the web service.

#### 6.5.3.2 [void BrisaWebServiceProvider::addFile](#) (const QString *path*, QString *filePath*)

Call this method to add a [BrisaWebFile](#) to the web service.

Referenced by [BrisaUpnp::BrisaService::buildWebServiceTree\(\)](#), and [BrisaCore::BrisaWebservice::publishFile\(\)](#).

#### 6.5.3.3 [void BrisaCore::BrisaWebServiceProvider::indexRequested](#) ([QxtWebRequestEvent](#) \* *event*) [inline, protected]

Referenced by [pageRequestedEvent\(\)](#).

#### 6.5.3.4 [void BrisaWebServiceProvider::pageRequestedEvent](#) ([QxtWebRequestEvent](#) \* *event*)

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

- [src/core/brisawebserver.h](#)
- [src/core/brisawebserver.cpp](#)

## 6.6 BrisaCore::BrisaWebStaticContent Class Reference

The [BrisaWebStaticContent](#) class stores a QString into the web server.

```
#include <BrisaCore/BrisaWebStaticContent>
```

### Public Slots

- void [index](#) (QxtWebRequestEvent \*event)

### Public Member Functions

- [BrisaWebStaticContent](#) (QxtAbstractWebSessionManager \*sm, QString content, QObject \*parent=0)

*Constructor for BrisaWebStaticContent.*

- [~BrisaWebStaticContent](#) ()

*Destructor for BrisaWebStaticContent.*

### 6.6.1 Detailed Description

The [BrisaWebStaticContent](#) class stores a QString into the web server. Use this class to store static content in the web server using a string format.

### 6.6.2 Constructor & Destructor Documentation

#### 6.6.2.1 BrisaWebStaticContent::BrisaWebStaticContent (QxtAbstractWebSessionManager \* *sm*, QString *content*, QObject \* *parent* = 0) [inline]

Constructor for [BrisaWebStaticContent](#). Stores the given QString.

#### 6.6.2.2 BrisaWebStaticContent::~BrisaWebStaticContent () [inline]

Destructor for [BrisaWebStaticContent](#).

### 6.6.3 Member Function Documentation

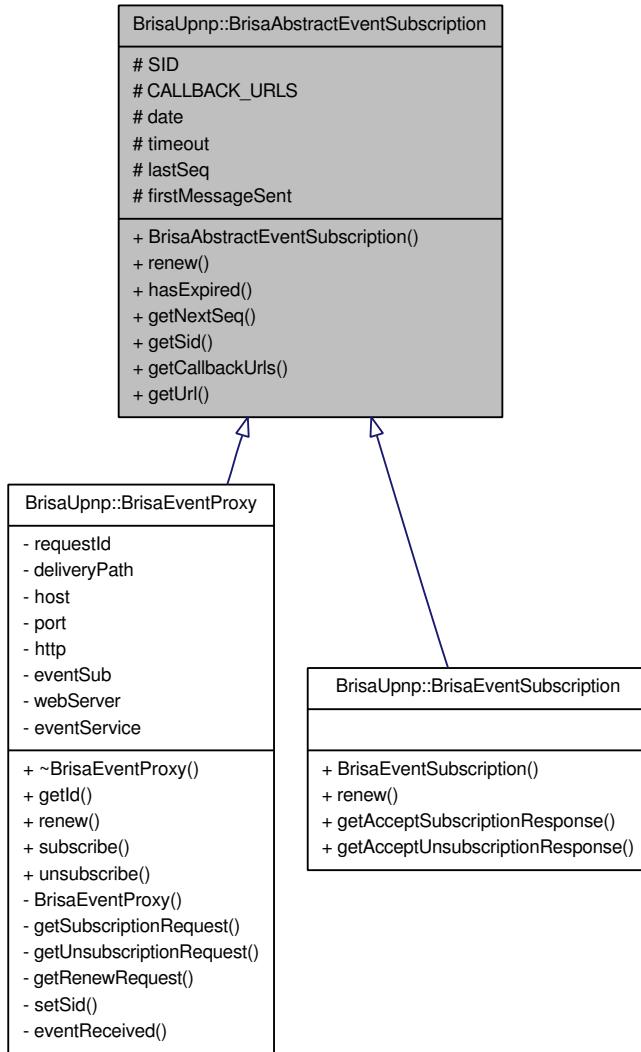
#### 6.6.3.1 void BrisaCore::BrisaWebStaticContent::index (QxtWebRequestEvent \* *event*) [inline, slot]

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

- src/core/[brisawebserver.h](#)
- src/core/[brisawebserver.cpp](#)

## 6.7 BrisaUpnp::BrisaAbstractEventSubscription Class Reference

#include <brisaabstracteventsubscription.h> Inheritance diagram for BrisaUpnp::BrisaAbstractEventSubscription:



### Public Member Functions

- **BrisaAbstractEventSubscription** (const QString &sid, const QStringList &callbackUrls, const int &timeout=-1, QObject \*parent=0)
 

*Constructs an abstract event subscription with given sid, list of callbackUrls, timeout and parent.*
- virtual void **renew** (const int &newTimeout=-1)=0
 

*Renews the subscription for the given newTimeout.*
- bool **hasExpired** () const
 

*Checks if the subscription has already expired.*

- quint32 [getNextSeq \(\)](#)  
*Returns the next event key for this subscription.*
- QString [getSid \(\) const](#)  
*Returns this subscription's SID.*
- QStringList [getCallbackUrls \(\) const](#)  
*Returns this subscription's list of callback URLs.*
- QUrl [getUrl \(\)](#)  
*Returns this subscription's first callback URL.*

## Protected Attributes

- QString [SID](#)
- const QStringList [CALLBACK\\_URLS](#)
- QDateTime [date](#)
- int [timeout](#)
- quint32 [lastSeq](#)
- bool [firstMessageSent](#)

### 6.7.1 Constructor & Destructor Documentation

#### 6.7.1.1 BrisaAbstractEventSubscription::BrisaAbstractEventSubscription (const QString & *sid*, const QStringList & *callbackUrls*, const int & *timeout* = -1, QObject \* *parent* = 0)

Constructs an abstract event subscription with given *sid*, list of *callbackUrls*, *timeout* and *parent*. *timeout* less than 0 means infinite.

### 6.7.2 Member Function Documentation

#### 6.7.2.1 QStringList BrisaAbstractEventSubscription::getCallbackUrls () const

Returns this subscription's list of callback URLs.

Referenced by BrisaUpnp::BrisaEventMessage::getMessageHeader().

#### 6.7.2.2 quint32 BrisaAbstractEventSubscription::getNextSeq ()

Returns the next event key for this subscription.

#### 6.7.2.3 QString BrisaAbstractEventSubscription::getSid () const

Returns this subscription's SID.

Referenced by BrisaUpnp::BrisaEventMessage::getMessageHeader(), and BrisaUpnp::BrisaEventController::subscribe().

#### 6.7.2.4 **QUrl BrisaAbstractEventSubscription::getUrl ()**

Returns this subscription's first callback URL.

Referenced by BrisaUpnp::BrisaEventController::subscribe().

#### 6.7.2.5 **bool BrisaAbstractEventSubscription::hasExpired () const**

Checks if the subscription has already expired. Returns true if it has expired, else returns false.

#### 6.7.2.6 **void BrisaAbstractEventSubscription::renew (const int & newTimeout = -1) [pure virtual]**

Renews the subscription for the given *newTimeout*.

Implemented in [BrisaUpnp::BrisaEventProxy](#), and [BrisaUpnp::BrisaEventSubscription](#).

### 6.7.3 Member Data Documentation

#### 6.7.3.1 **const QStringList BrisaUpnp::BrisaAbstractEventSubscription::CALLBACK\_URLS [protected]**

Referenced by getCallbackUrls(), and getUrl().

#### 6.7.3.2 **QDateTime BrisaUpnp::BrisaAbstractEventSubscription::date [protected]**

Referenced by hasExpired(), BrisaUpnp::BrisaEventSubscription::renew(), and renew().

#### 6.7.3.3 **bool BrisaUpnp::BrisaAbstractEventSubscription::firstMessageSent [protected]**

Referenced by getNextSeq().

#### 6.7.3.4 **quint32 BrisaUpnp::BrisaAbstractEventSubscription::lastSeq [protected]**

Referenced by getNextSeq().

#### 6.7.3.5 **QString BrisaUpnp::BrisaAbstractEventSubscription::SID [protected]**

Referenced by BrisaUpnp::BrisaEventSubscription::getAcceptSubscriptionResponse(), and getSid().

#### 6.7.3.6 **int BrisaUpnp::BrisaAbstractEventSubscription::timeout [protected]**

Referenced by BrisaUpnp::BrisaEventSubscription::getAcceptSubscriptionResponse(), hasExpired(), BrisaUpnp::BrisaEventSubscription::renew(), and renew().

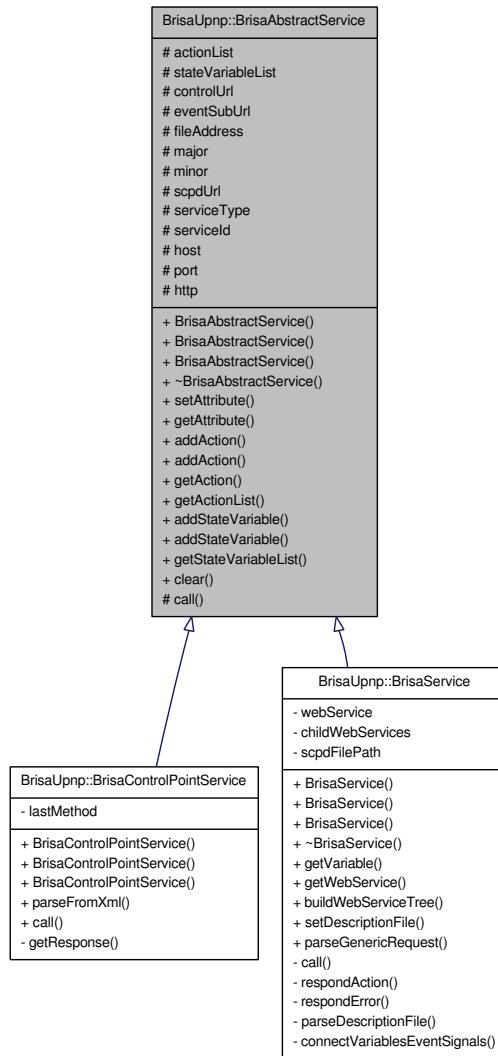
The documentation for this class was generated from the following files:

- [src/upnp/brisabSTRACTEVENTSUBSCRIPTION.h](#)
- [src/upnp/brisabSTRACTEVENTSUBSCRIPTION.cpp](#)

## 6.8 BrisaUpnp::BrisaAbstractService Class Reference

```
#include <brisaabstractservice.h>Inheritance
BrisaUpnp::BrisaAbstractService:
```

diagram for



### Public Types

- enum `xmlTags` {
 Major, Minor, FileAddress, ServiceType,  
 ServiceId, ScpdUrl, ControlUrl, EventSubUrl,  
 Host, Port }

### Signals

- void `requestFinished` (QString root, QString lastMethod)

## Public Member Functions

- `BrisaAbstractService (QObject *parent=0)`
- `BrisaAbstractService (const QString &serviceType, const QString &serviceId="", const QString &scpdUrl="", const QString &controlUrl="", const QString &eventSubUrl="", const QString &host="", QObject *parent=0)`
- `BrisaAbstractService (BrisaAbstractService &service)`
- `virtual ~BrisaAbstractService ()`
- `void setAttribute (xmlTags key, const QString &value)`
- `QString getAttribute (xmlTags key)`
- `void addAction (const QString &name)`
- `void addAction (BrisaAction *action)`
- `BrisaAction * getAction (const QString &name)`
- `QList< BrisaAction * > getActionList ()`
- `void addStateVariable (BrisaStateVariable *stateVariable)`
- `void addStateVariable (const QString &sendEvents, const QString &name, const QString &datatype, const QString &defaultValue, const QString &maximum, const QString &minimum, const QString &step)`
- `const QList< BrisaStateVariable * > getStateVariableList ()`
- `void clear ()`

## Protected Member Functions

- `virtual void call (const QString &method, const QMap< QString, QString > &param)=0`

## Protected Attributes

- `QList< BrisaAction * > actionList`
- `QList< BrisaStateVariable * > stateVariableList`
- `QString controlUrl`
- `QString eventSubUrl`
- `QString fileAddress`
- `QString major`
- `QString minor`
- `QString scpdUrl`
- `QString serviceType`
- `QString serviceId`
- `QString host`
- `int port`
- `QtSoapHttpTransport http`

### 6.8.1 Member Enumeration Documentation

#### 6.8.1.1 enum BrisaUpnp::BrisaAbstractService::xmlTags

Enumerator:

*Major*

*Minor*

*FileAddress*

*ServiceType*

*ServiceId*

*ScpdUrl*

*ControlUrl*

*EventSubUrl*

*Host*

*Port*

## 6.8.2 Constructor & Destructor Documentation

**6.8.2.1 BrisaAbstractService::BrisaAbstractService (QObject \* *parent* = 0)**

**6.8.2.2 BrisaAbstractService::BrisaAbstractService (const QString & *serviceType*, const QString & *serviceId* = "", const QString & *scpdUrl* = "", const QString & *controlUrl* = "", const QString & *eventSubUrl* = "", const QString & *host* = "", QObject \* *parent* = 0)**

**6.8.2.3 BrisaAbstractService::BrisaAbstractService (BrisaAbstractService & *service*)**

**6.8.2.4 BrisaAbstractService::~BrisaAbstractService () [virtual]**

## 6.8.3 Member Function Documentation

**6.8.3.1 void BrisaAbstractService::addAction (BrisaAction \* *action*)**

**6.8.3.2 void BrisaAbstractService::addAction (const QString & *name*)**

Referenced by BrisaUpnp::BrisaServiceXMLHandler::endElement().

**6.8.3.3 void BrisaAbstractService::addStateVariable (const QString & *sendEvents*, const QString & *name*, const QString & *datatype*, const QString & *defaultValue*, const QString & *maximum*, const QString & *minimum*, const QString & *step*)**

**6.8.3.4 void BrisaAbstractService::addStateVariable (BrisaStateVariable \* *stateVariable*)**

Referenced by addStateVariable(), and BrisaUpnp::BrisaServiceXMLHandler::endElement().

**6.8.3.5 virtual void BrisaUpnp::BrisaAbstractService::call (const QString & *method*, const QMap<QString, QString> & *param*) [protected, pure virtual]**

Implemented in [BrisaUpnp::BrisaControlPointService](#).

**6.8.3.6 void BrisaAbstractService::clear ()**

**6.8.3.7 BrisaAction \* BrisaAbstractService::getAction (const QString & *name*)**

Referenced by addAction(), and BrisaUpnp::BrisaServiceXMLHandler::endElement().

### 6.8.3.8 `QList< BrisaAction * > BrisaAbstractService::getActionList ()`

Referenced by `BrisaAbstractService()`.

### 6.8.3.9 `QString BrisaAbstractService::getAttribute (xmlTags key)`

Referenced by `BrisaAbstractService()`, `BrisaUpnp::BrisaDeviceXMLHandlerCP::endElement()`, `BrisaUpnp::BrisaDevice::getServiceById()`, `BrisaUpnp::BrisaDevice::getServiceByType()`, and `BrisaUpnp::BrisaControlPoint::getSubscriptionProxy()`.

### 6.8.3.10 `const QList< BrisaStateVariable * > BrisaAbstractService::getStateVariableList ()`

### 6.8.3.11 `void BrisaUpnp::BrisaAbstractService::requestFinished (QString root, QString lastMethod) [signal]`

### 6.8.3.12 `void BrisaAbstractService::setAttribute (xmlTags key, const QString & value)`

Referenced by `BrisaUpnp::BrisaDeviceXMLHandlerCP::characters()` and `BrisaUpnp::BrisaServiceXMLHandler::characters()`.

## 6.8.4 Member Data Documentation

### 6.8.4.1 `QList<BrisaAction *> BrisaUpnp::BrisaAbstractService::actionList [protected]`

Referenced by `addAction()`, `BrisaAbstractService()`, `getAction()`, `getActionList()`, and `~BrisaAbstractService()`.

### 6.8.4.2 `QString BrisaUpnp::BrisaAbstractService::controlUrl [protected]`

Referenced by `BrisaUpnp::BrisaService::buildWebServiceTree()`, `BrisaUpnp::BrisaControlPointService::call()`, `clear()`, `getAttribute()`, and `setAttribute()`.

### 6.8.4.3 `QString BrisaUpnp::BrisaAbstractService::eventSubUrl [protected]`

Referenced by `BrisaUpnp::BrisaService::buildWebServiceTree()`, `clear()`, `getAttribute()`, and `setAttribute()`.

### 6.8.4.4 `QString BrisaUpnp::BrisaAbstractService::fileAddress [protected]`

Referenced by `clear()`, `getAttribute()`, and `setAttribute()`.

### 6.8.4.5 `QString BrisaUpnp::BrisaAbstractService::host [protected]`

Referenced by `clear()`, `getAttribute()`, and `setAttribute()`.

**6.8.4.6 QtSoapHttpTransport BrisaUpnp::BrisaAbstractService::http [protected]**

Referenced by BrisaAbstractService(), BrisaUpnp::BrisaControlPointService::BrisaControlPointService(), BrisaUpnp::BrisaControlPointService::call(), and setAttribute().

**6.8.4.7 QString BrisaUpnp::BrisaAbstractService::major [protected]**

Referenced by BrisaAbstractService(), clear(), getAttribute(), and setAttribute().

**6.8.4.8 QString BrisaUpnp::BrisaAbstractService::minor [protected]**

Referenced by BrisaAbstractService(), clear(), getAttribute(), and setAttribute().

**6.8.4.9 int BrisaUpnp::BrisaAbstractService::port [protected]**

Referenced by BrisaAbstractService(), clear(), getAttribute(), and setAttribute().

**6.8.4.10 QString BrisaUpnp::BrisaAbstractService::scpdUrl [protected]**

Referenced by BrisaUpnp::BrisaService::buildWebServiceTree(), clear(), getAttribute(), and setAttribute().

**6.8.4.11 QString BrisaUpnp::BrisaAbstractService::serviceId [protected]**

Referenced by clear(), getAttribute(), and setAttribute().

**6.8.4.12 QString BrisaUpnp::BrisaAbstractService::serviceType [protected]**

Referenced by BrisaUpnp::BrisaControlPointService::call(), clear(), getAttribute(), BrisaUpnp::BrisaService::parseGenericRequest(), and setAttribute().

**6.8.4.13 QList<BrisaStateVariable \*> BrisaUpnp::BrisaAbstractService::stateVariableList [protected]**

Referenced by addStateVariable(), BrisaUpnp::BrisaService::buildWebServiceTree(), getStateVariableList(), BrisaUpnp::BrisaService::getVariable(), and ~BrisaAbstractService().

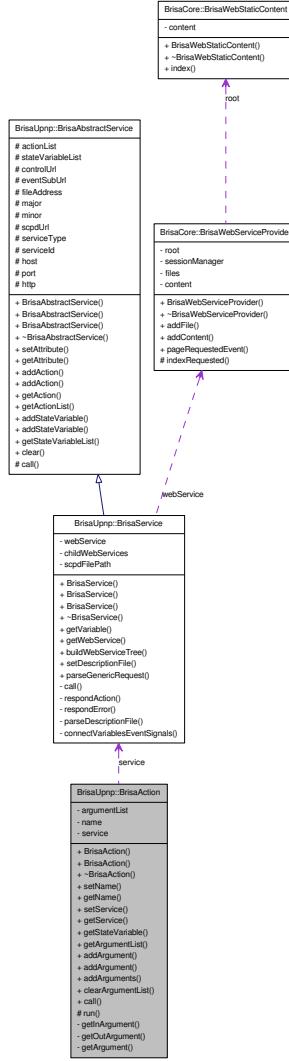
The documentation for this class was generated from the following files:

- src/upnp/[brisabSTRACTSERVICE.h](#)
- src/upnp/[brisabSTRACTSERVICE.cpp](#)

## 6.9 BrisaUpnp::BrisaAction Class Reference

Template method class that represents each service's action.

#include <BrisaUpnp/BrisaAction>Collaboration diagram for BrisaUpnp::BrisaAction:



### Public Member Functions

- **BrisaAction (QString name="", BrisaService \*service=0, QObject \*parent=0)**  
*Constructs an action with given name, parent and service that it is related to.*
- **BrisaAction (const BrisaAction &action)**  
*Constructs a new action based on action.*
- **virtual ~BrisaAction ()**  
*Destroys the action.*

- void [setName \(QString name\)](#)  
*Sets a new name to the action.*
- QString [getName \(\) const](#)  
*Returns the action's name.*
- void [setService \(BrisaService \\*service\)](#)  
*Sets a new service for this action.*
- BrisaService \* [getService \(\) const](#)  
*Returns the service that this action is related to.*
- BrisaStateVariable \* [getStateVariable \(const QString &name\) const](#)  
*Returns the related service's state variable with the given name.*
- QList< BrisaArgument \* > [getArgumentList \(\) const](#)  
*Returns this action's list of arguments.*
- void [addArgument \(QString name="", QString direction="", QString relatedStateVariable ""\)](#)  
*Adds an argument with given name, direction and relatedStateVariable to this action's list of arguments.*
- void [addArgument \(BrisaArgument \\*argumentA\)](#)  
*Adds given argument to this action's list of arguments.*
- void [addArguments \(const QList< BrisaArgument \\* > argumentsA\)](#)  
*Adds given list of arguments to this action's list of arguments.*
- void [clearArgumentList \(\)](#)  
*Clears this action's argument list.*
- bool [call \(const QMap< QString, QString > &inArguments, QMap< QString, QString > &outArguments\)](#)  
*Validates inArguments, outArguments and runs the action.*

## Protected Member Functions

- virtual QMap< QString, QString > [run \(const QMap< QString, QString > &inArguments\)](#)  
*The actual action.*

### 6.9.1 Detailed Description

Template method class that represents each service's action. Create a class derived from [BrisaAction](#) and reimplement the method [run\(\)](#). That method is the one which is called when this action is invoked and defines the action's behavior.

The method [run\(\)](#) receives a `QMap<QString, QString>` with the input arguments. They are organized as key -- the argument name -- and value -- the argument value. It has to return a `QMap<QString, QString>` with the output arguments organized the same way.

If any of the returned output arguments is not defined in the service description file, [BrisaAction](#) will show an error message at the debug output stream and send an error message to the control point.

## 6.9.2 Constructor & Destructor Documentation

### 6.9.2.1 BrisaAction::BrisaAction (QString *name* = "", BrisaService \* *service* = 0, QObject \* *parent* = 0)

Constructs an action with given *name*, *parent* and *service* that it is related to.

### 6.9.2.2 BrisaAction::BrisaAction (const BrisaAction & *action*)

Constructs a new action based on *action*.

### 6.9.2.3 BrisaAction::~BrisaAction () [virtual]

Destroys the action. It has to be overridden for properly destroying the derived actions when necessary.

## 6.9.3 Member Function Documentation

### 6.9.3.1 void BrisaAction::addArgument (BrisaArgument \* *argumentA*)

Adds given *argument* to this action's list of arguments.

### 6.9.3.2 void BrisaAction::addArgument (QString *name* = "", QString *direction* = "", QString *relatedStateVariable* = "")

Adds an argument with given *name*, *direction* and *relatedStateVariable* to this action's list of arguments.

Referenced by addArguments(), and BrisaUpnp::BrisaServiceXMLHandler::endElement().

### 6.9.3.3 void BrisaAction::addArguments (const QList< BrisaArgument \* > *argumentsA*)

Adds given list of *arguments* to this action's list of arguments.

Referenced by BrisaUpnp::BrisaServiceXMLHandler::endElement().

### 6.9.3.4 bool BrisaAction::call (const QMap< QString, QString > & *inArguments*, QMap< QString, QString > & *outArguments*)

Validates *inArguments*, *outArguments* and runs the action. *outArguments* is an output parameter. This method returns true in case of successful running of the action, else returns false.

### 6.9.3.5 void BrisaAction::clearArgumentList ()

Clears this action's argument list.

Referenced by BrisaUpnp::BrisaServiceXMLHandler::endElement().

**6.9.3.6 QList< BrisaArgument \* > BrisaAction::getArgumentList () const**

Returns this action's list of arguments.

Referenced by BrisaAction(), and BrisaUpnp::BrisaServiceXMLHandler::endElement().

**6.9.3.7 QString BrisaAction::getName () const**

Returns the action's name.

Referenced by BrisaUpnp::BrisaAbstractService::addAction(),  
BrisaUpnp::BrisaServiceXMLHandler::endElement(), and getStateVariable(). call(),

**6.9.3.8 BrisaService \* BrisaAction::getService () const**

Returns the service that this action is related to.

Referenced by getStateVariable().

**6.9.3.9 BrisaStateVariable \* BrisaAction::getStateVariable (const QString & *name*) const**

Returns the related service's state variable with the given *name*. If it cannot find its related service or the state variable, then it returns a null pointer.

**6.9.3.10 QMap< QString, QString > BrisaAction::run (const QMap< QString, QString > & *inArguments*) [protected, virtual]**

The actual action. This method must be reimplemented for each user's action. It receives *inArguments* as input arguments and must return the right output arguments as described in the service's description file.

Referenced by call().

**6.9.3.11 void BrisaAction::setName (QString *name*)**

Sets a new name to the action.

Referenced by BrisaUpnp::BrisaServiceXMLHandler::characters().

**6.9.3.12 void BrisaAction::setService (BrisaService \* *service*)**

Sets a new service for this action.

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

- src/upnp/[brisaction.h](#)
- src/upnp/[brisaction.cpp](#)

## 6.10 BrisaUpnp::BrisaActionXmlParser Class Reference

XML parser for SOAP requests.

```
#include <BrisaUpnp/BrisaActionXmlParser>
```

### Public Member Functions

- [BrisaActionXmlParser \(\)](#)

*Constructor.*

- virtual [~BrisaActionXmlParser \(\)](#)

*Destructor.*

- [bool parseSOAP \(\)](#)

*Call this method to parse the SOAP request set by the setXmlContent method.*

- [void parseElement \(QDomElement &element\)](#)

- [void setXmlContent \(const QByteArray &content\)](#)

*Sets the content to be parsed.*

### Public Attributes

- [QMap< QString, QString > args](#)
- [QString method](#)
- [QString serviceType](#)

#### 6.10.1 Detailed Description

XML parser for SOAP requests. [BrisaActionXmlParser](#) parses information coming from the webserver. If a action is detected, public class members args, method, serviceType will be filled with the parsed data.

[BrisaActionXmlParser](#) uses DOM.

#### 6.10.2 Constructor & Destructor Documentation

##### 6.10.2.1 BrisaActionXmlParser::BrisaActionXmlParser ()

Constructor.

##### 6.10.2.2 BrisaActionXmlParser::~BrisaActionXmlParser () [virtual]

Destructor.

### 6.10.3 Member Function Documentation

#### 6.10.3.1 void BrisaActionXmlParser::parseElement (QDomElement & *element*)

Referenced by parseSOAP().

#### 6.10.3.2 bool BrisaActionXmlParser::parseSOAP ()

Call this method to parse the SOAP request set by the setXmlContent method.

Referenced by BrisaUpnp::BrisaService::parseGenericRequest().

#### 6.10.3.3 void BrisaActionXmlParser::setXmlContent (const QByteArray & *content*)

Sets the content to be parsed.

Referenced by BrisaUpnp::BrisaService::parseGenericRequest().

### 6.10.4 Member Data Documentation

#### 6.10.4.1 QMap<QString, QString> BrisaUpnp::BrisaActionXmlParser::args

Referenced by parseElement(), and BrisaUpnp::BrisaService::parseGenericRequest().

#### 6.10.4.2 QString BrisaUpnp::BrisaActionXmlParser::method

Referenced by parseElement(), BrisaUpnp::BrisaService::parseGenericRequest(), and parseSOAP().

#### 6.10.4.3 QString BrisaUpnp::BrisaActionXmlParser::serviceType

Referenced by parseElement(), BrisaUpnp::BrisaService::parseGenericRequest(), and parseSOAP().

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

- [src/upnp/device/brisactionxmlparser.h](#)
- [src/upnp/device/brisactionxmlparser.cpp](#)

## 6.11 BrisaUpnp::BrisaArgument Class Reference

```
#include <brisaargument.h>
```

### Public Types

- enum [xmlArgument](#) { [ArgumentName](#), [Direction](#), [RelatedStateVariable](#) }

### Public Member Functions

- [BrisaArgument](#) (const QString &name="", const QString &direction="", const QString &relatedStateVariable "")
- void [setAttribute](#) ([xmlArgument](#) key, const QString &value)
- QString [getAttribute](#) ([xmlArgument](#) key) const
- void [clear](#) ()

#### 6.11.1 Member Enumeration Documentation

##### 6.11.1.1 enum BrisaUpnp::BrisaArgument::xmlArgument

Enumerator:

*ArgumentName*  
*Direction*  
*RelatedStateVariable*

#### 6.11.2 Constructor & Destructor Documentation

##### 6.11.2.1 BrisaArgument::BrisaArgument (const QString & name = "", const QString & direction = "", const QString & relatedStateVariable = "")

#### 6.11.3 Member Function Documentation

##### 6.11.3.1 void BrisaArgument::clear ()

##### 6.11.3.2 QString BrisaArgument::getAttribute (xmlArgument key) const

##### 6.11.3.3 void BrisaArgument::setAttribute (xmlArgument key, const QString & value)

Referenced by BrisaUpnp::BrisaServiceXMLHandler::characters().

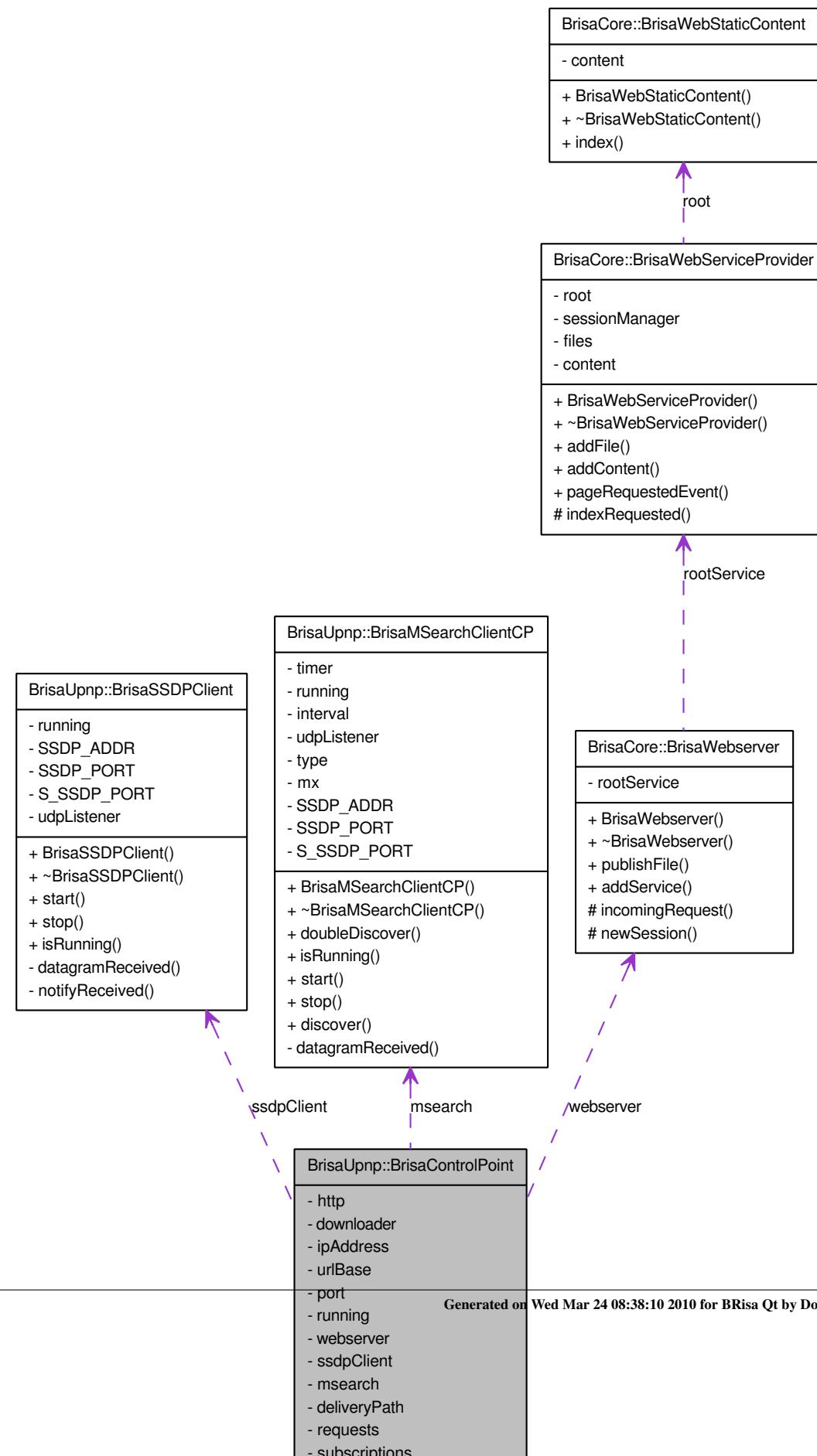
The documentation for this class was generated from the following files:

- src/upnp/[brisargument.h](#)
- src/upnp/[brisargument.cpp](#)

## 6.12 BrisaUpnp::BrisaControlPoint Class Reference

Class that implements the control part in UPnP Architecture.

#include <BrisaUpnp/BrisaControlPoint>Collaboration diagram for BrisaUpnp::BrisaControlPoint:



## Signals

- void **deviceFound** (BrisaControlPointDevice \*device)

*This signal is emitted when a new device is found in network and all its attributes are created by the xml reading.*

- void **deviceGone** (QString usn)

*This signal is emitted when a device leaves the network, that means that the ssdp client received a "ssdp:byebye" message from the device and, to handle this, the control point emit a deviceGone event with the device's usn as parameter.*

## Public Member Functions

- **BrisaControlPoint** (QObject \*parent=0, QString st="ssdp:all", int mx=5)

*Constructs a BrisaControlPoint with the given parent, the service type st and the passed interval mx.*

- **~BrisaControlPoint** ()

*Destructor of BrisaControlPoint class.*

- void **start** ()

*This function starts the control point, the ssdpClient and the msearch.*

- void **stop** ()

*This function stops the control point, the ssdpClient and the msearch.*

- bool **isRunning** ()

*Returns true if the control point is running.*

- void **discover** ()

*Starts the control point msearch discover.*

- **BrisaEventProxy** \* **getSubscriptionProxy** (BrisaControlPointService \*service)

*Function to get a event proxy to subscribe, unsubscribe or renew the events from a service.*

### 6.12.1 Detailed Description

Class that implements the control part in UPnP Architecture. Create a ControlPoint and **start()**, then **discover()** devices will be found in network. If you don't want to look for more devices then use **stop()**.

### 6.12.2 Constructor & Destructor Documentation

#### 6.12.2.1 BrisaControlPoint::BrisaControlPoint (QObject \*parent = 0, QString st = "ssdp:all", int mx = 5)

Constructs a BrisaControlPoint with the given *parent*, the service type *st* and the passed interval *mx*.

### 6.12.2.2 BrisaControlPoint::~BrisaControlPoint ()

Destructor of [BrisaControlPoint](#) class.

## 6.12.3 Member Function Documentation

### 6.12.3.1 void BrisaControlPoint::deviceFound (BrisaControlPointDevice \* *device*) [signal]

This signal is emitted when a new device is find in network and all it's attributes are created by the xml reading.

See also:

[deviceGone\(QString usn\)](#)

Referenced by [BrisaControlPoint\(\)](#).

### 6.12.3.2 void BrisaControlPoint::deviceGone (QString *usn*) [signal]

This signal is emitted when a device leaves the network, that means that the the ssdp client received a "ssdp:byebye" message from the device and, to handle this, the control point emit a deviceGone event with the device's *usn* as parameter.

See also:

[deviceFound\(BrisaControlPointDevice \\*device\)](#)

### 6.12.3.3 void BrisaControlPoint::discover ()

Starts the control point msearch discover.

### 6.12.3.4 BrisaEventProxy \* BrisaControlPoint::getSubscriptionProxy (BrisaControlPointService \* *service*)

Function to get a event proxy to subscribe, usubscribe or renew the events from a *service*.

### 6.12.3.5 bool BrisaControlPoint::isRunning ()

Returns true if the control point is running.

See also:

[start\(\), stop\(\)](#)

Referenced by [start\(\)](#), [stop\(\)](#), and [~BrisaControlPoint\(\)](#).

### 6.12.3.6 void BrisaControlPoint::start ()

This function starts the control point, the ssdpClient and the msearch.

**See also:**

[stop\(\)](#), [isRunning\(\)](#)

#### 6.12.3.7 void BrisaControlPoint::stop ()

This function stops the control point, the ssdpClient and the msearch.

**See also:**

[start\(\)](#), [isRunning\(\)](#)

Referenced by [~BrisaControlPoint\(\)](#).

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

- src/upnp/controlpoint/[brisaccontrolpoint.h](#)
- src/upnp/controlpoint/[brisaccontrolpoint.cpp](#)

## 6.13 BrisaUpnp::BrisaControlPointDevice Class Reference

Class that implements the devices that control point part is going to handle.

```
#include <BrisaUpnp/BrisaControlPointDevice>
```

### Public Types

- enum `xmlTags` {
 Major, Minor, UrlBase, DeviceType,  
 FriendlyName, Manufacturer, ManufacturerUrl, ModelDescription,  
 ModelName, ModelNumber, ModelUrl, SerialNumber,  
 Udn, Upc, PresentationUrl, FileAddress }

*This enum specifies the devices attributes that are going to be set/get.*

### Public Member Functions

- `BrisaControlPointDevice (QObject *parent=0)`  
*Constructor to `BrisaControlPointDevice`, when it makes use of this constructor the device's attributes should be set.*
- `BrisaControlPointDevice (QTemporaryFile *xml, QUrl *url, QObject *parent=0)`  
*Constructor for `BrisaControlPointDevice` that receives a xml file containing the device description, so that the device's attributes can be initialized.*
- `BrisaControlPointDevice (BrisaControlPointDevice &dev, QObject *parent=0)`  
*Constructor for `BrisaControlPointDevice` that receives another object of the same type and copy it's attributes.*
- `BrisaControlPointDevice (QString deviceType, QString friendlyName="", QString manufacturer="", QString manufacturerURL="", QString modelDescription="", QString modelName="", QString modelNumber="", QString modelURL="", QString serialNumber="", QString UDN="", QString UPC="", QString presentationURL="", QObject *parent=0)`  
*Constructor where all device's attributes are passed as parameter.*
- `~BrisaControlPointDevice ()`  
*Destructor.*
- `QString getAttribute (xmlTags key)`  
*Gets the key attribute.*
- `void setAttribute (xmlTags key, QString value)`  
*Set a device's attribute, depending on the key that is passed as parameter, so that, the attribute e set to value v.*
- `void addIcon (BrisaIcon *icon)`  
*Add icon to device's icon list.*

- void [addService \(BrisaControlPointService \\*service\)](#)  
*Add serviceSwap to device's service list.*
- void [addDevice \(BrisaControlPointDevice \\*device\)](#)  
*Add device to device's embedded device list.*
- QList< [BrisaIcon \\* > getIconList \(\)](#)  
*Returns device's icon list.*
- QList< [BrisaControlPointService \\* > & getServiceList \(\)](#)  
*Returns device's service list.*
- QList< [BrisaControlPointDevice \\* > getEmbeddedDeviceList \(\)](#)  
*Returns device's embedded device list.*
- [BrisaControlPointService \\* getServiceById \(QString serviceId\)](#)  
*Check the device's service list and return the service that has the passed .*
- [BrisaControlPointService \\* getServiceByType \(QString serviceType\)](#)  
*Check the device's service list and return the service that has the passed .*
- void [clear \(\)](#)  
*Clear device's attributes.*

### 6.13.1 Detailed Description

Class that implements the devices that control point part is going to handle.

### 6.13.2 Member Enumeration Documentation

#### 6.13.2.1 enum BrisaUpnp::BrisaControlPointDevice::xmlTags

This enum specifies the devices attributes that are going to be set/get.

##### Parameters:

- Major** Major version of the UPnP Device Architecture.
- Minor** Minor version of the UPnP Device Architecture.
- UrlBase** Defines the base URL. Used to construct fully-qualified URLs.
- DeviceType** UPnP device type. Single URI.
- FriendlyName** Short description for end user.
- Manufacturer** Manufacturer's name.
- ManufacturerUrl** Web site for Manufacturer.
- ModelDescription** Long description for end user.
- ModelName** Model name. Should be less then 32 characters.
- ModelNumber** Model number. Should be less then 32 characters.
- ModelUrl** Web site for model.

*SerialNumber* Serial number. Should be less then 64 characters

*Udn* Unique Device Name. Universally-unique identifier for the device.

*Upc* Universal Product Code. 12-digit, all-numeric code that identifies the consumer package.

*PresentationUrl* URL to presentation for device.

*FileAddress* Device's file address.

#### See also:

[setAttribute\(xmlTags key, QString v\)](#), [getAttribute\(xmlTags key\)](#)

#### Enumerator:

*Major*

*Minor*

*UrlBase*

*DeviceType*

*FriendlyName*

*Manufacturer*

*ManufacturerUrl*

*ModelDescription*

*ModelName*

*ModelNumber*

*ModelUrl*

*SerialNumber*

*Udn*

*Upc*

*PresentationUrl*

*FileAddress*

### 6.13.3 Constructor & Destructor Documentation

#### 6.13.3.1 BrisaControlPointDevice::BrisaControlPointDevice (*QObject \*parent = 0*)

Constructor to [BrisaControlPointDevice](#), when it makes use of this constructor the device's attributes should be set.

#### 6.13.3.2 BrisaControlPointDevice::BrisaControlPointDevice (*QTemporaryFile \*xml*, *QUrl \*url*, *QObject \*parent = 0*)

Constructor for [BrisaControlPointDevice](#) that receives a xml file containing the device description, so that the device's attributes can be initialized.

#### 6.13.3.3 BrisaControlPointDevice::BrisaControlPointDevice (*BrisaControlPointDevice & dev*, *QObject \*parent = 0*)

Constructor for [BrisaControlPointDevice](#) that receives another object of the same type and copy it's attributes.

**6.13.3.4 Brisacontrolpointdevice::BrisaControlPointDevice (QString *deviceType*, QString *friendlyName* = "", QString *manufacturer* = "", QString *manufacturerURL* = "", QString *modelDescription* = "", QString *modelName* = "", QString *modelNumber* = "", QString *modelURL* = "", QString *serialNumber* = "", QString *UDN* = "", QString *UPC* = "", QString *presentationURL* = "", QObject \**parent* = 0)**

Constructor where all device's attributes are passed as parameter.

**6.13.3.5 Brisacontrolpointdevice::~BrisaControlPointDevice ()**

Destructor.

#### 6.13.4 Member Function Documentation

**6.13.4.1 void Brisacontrolpointdevice::addDevice (BrisaControlPointDevice \**device*)**

Add *device* to device's embedded device list.

Referenced by BrisaUpnp::BrisaDeviceXMLHandlerCP::EndElement().

**6.13.4.2 void Brisacontrolpointdevice::addIcon (BrisaIcon \**icon*)**

Add *icon* to device's icon list.

Referenced by BrisaUpnp::BrisaDeviceXMLHandlerCP::EndElement().

**6.13.4.3 void Brisacontrolpointdevice::addService (BrisaControlPointService \**service*)**

Add *service* to device's service list.

Referenced by BrisaUpnp::BrisaDeviceXMLHandlerCP::EndElement().

**6.13.4.4 void Brisacontrolpointdevice::clear ()**

Clear device's attributes.

**6.13.4.5 QString Brisacontrolpointdevice::getAttribute (xmlTags *key*)**

Gets the *key* attribute.

Referenced by BrisaControlPointDevice(), and BrisaUpnp::BrisaDeviceXMLHandlerCP::EndElement().

**6.13.4.6 QList< Brisacontrolpointdevice \* > Brisacontrolpointdevice::getEmbeddedDeviceList  
0**

Returns device's embedded device list.

Referenced by BrisaControlPointDevice().

**6.13.4.7 QList< BrisaIcon \* > BrisaControlPointDevice::getIconList ()**

Returns device's icon list.

Referenced by BrisaControlPointDevice().

**6.13.4.8 BrisaControlPointService \* BrisaControlPointDevice::getServiceById (QString *serviceId*)**

Check the device's service list and return the service that has the passed .

**6.13.4.9 BrisaControlPointService \* BrisaControlPointDevice::getServiceByType (QString *serviceType*)**

Check the device's service list and return the service that has the passed .

**6.13.4.10 QList< BrisaControlPointService \* > & BrisaControlPointDevice::getServiceList ()**

Returns device's service list.

Referenced by BrisaControlPointDevice().

**6.13.4.11 void BrisaControlPointDevice::setAttribute (xmlTags *key*, QString *value*)**

Set a device's attribute, depending on the *key* that is passed as parameter, so that, the attribute e set to value *v*.

Referenced by BrisaControlPointDevice(), and BrisaUpnp::BrisaDeviceXMLHandlerCP::characters().

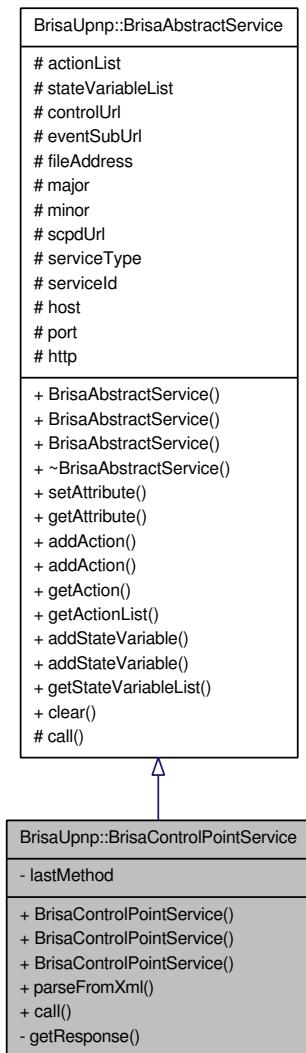
The documentation for this class was generated from the following files:

- src/upnp/controlpoint/[brisaccontrolpointdevice.h](#)
- src/upnp/controlpoint/[brisaccontrolpointdevice.cpp](#)

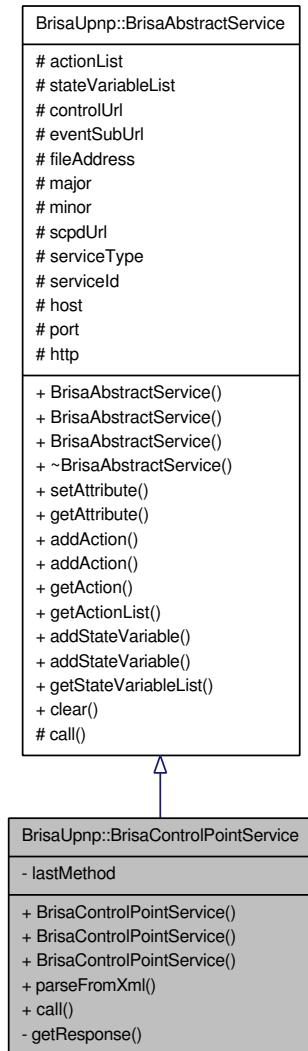
## 6.14 BrisaUpnp::BrisaControlPointService Class Reference

[BrisaControlPointService](#) is the class that implements action control in UPnP Architecture.

#include <BrisaUpnp/BrisaControlPointService> Inheritance diagram for BrisaUpnp::BrisaControlPointService:



Collaboration diagram for BrisaUpnp::BrisaControlPointService:



## Public Member Functions

- **`BrisaControlPointService`** (`QObject *parent=0`)
 

*Constructs an empty `BrisaControlPointService`.*
- **`BrisaControlPointService`** (`const QString &serviceType, const QString &serviceId="", const QString &scpdUrl="", const QString &controlUrl="", const QString &eventSubUrl="", const QString &host "", QObject *parent=0`)
 

*Constructs a `BrisaControlPointService` with the passed attributes.*
- **`BrisaControlPointService`** (`BrisaControlPointService &service`)
 

*Constructs a copy of serv.*
- **`void parseFromXml`** (`QTemporaryFile *xml`)

*Initializes the [BrisaControlPointService](#) from the parse of xml that is a xml description file containing the service information.*

- void **call** (const QString &*method*, const QMap<QString, QString> &*param*)  
*Call the method, with the passed param from a service, this action is performed by a webservice request.*

### 6.14.1 Detailed Description

[BrisaControlPointService](#) is the class that implements action control in UPnP Architecture. It performs the action requests it's used in control point part, so that the user can make action calls.

[BrisaControlPointService](#) is a [BrisaAbstractService](#).

### 6.14.2 Constructor & Destructor Documentation

#### 6.14.2.1 BrisaControlPointService::BrisaControlPointService (QObject \**parent* = 0)

Constructs an empty [BrisaControlPointService](#).

#### 6.14.2.2 BrisaControlPointService::BrisaControlPointService (const QString & *serviceType*, const QString & *serviceId* = "", const QString & *scpdUrl* = "", const QString & *controlUrl* = "", const QString & *eventSubUrl* = "", const QString & *host* = "", QObject \**parent* = 0)

Constructs a [BrisaControlPointService](#) with the passed attributes.

#### 6.14.2.3 BrisaControlPointService::BrisaControlPointService ([BrisaControlPointService](#) & *service*)

Constructs a copy of *serv*.

### 6.14.3 Member Function Documentation

#### 6.14.3.1 void BrisaControlPointService::call (const QString & *method*, const QMap<QString, QString> &*param*) [virtual]

Call the *method*, with the passed *param* from a service, this action is performed by a webservice request.

Implements [BrisaUpnp::BrisaAbstractService](#).

#### 6.14.3.2 void BrisaControlPointService::parseFromXml (QTemporaryFile \**xml*)

Initializes the [BrisaControlPointService](#) from the parse of *xml* that is a xml description file containing the service information.

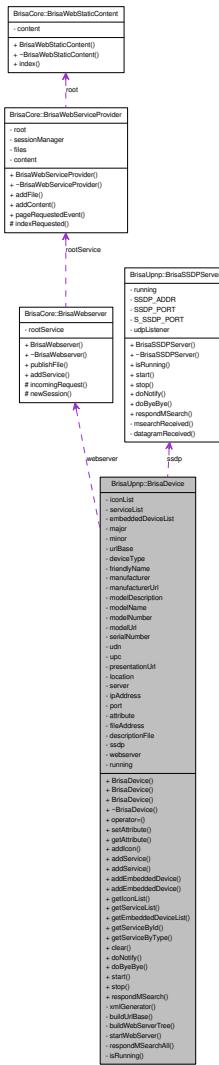
The documentation for this class was generated from the following files:

- src/upnp/controlpoint/[brisacontrolpointservice.h](#)
- src/upnp/controlpoint/[brisacontrolpointservice.cpp](#)

## 6.15 BrisaUpnp::BrisaDevice Class Reference

UPnP device implementation.

#include <BrisaUpnp/BrisaDevice>Collaboration diagram for BrisaUpnp::BrisaDevice:



## Public Types

- enum [xmlTags](#) {
- [Major](#), [Minor](#), [UrlBase](#), [DeviceType](#),
- [FriendlyName](#), [Manufacturer](#), [ManufacturerUrl](#), [ModelDescription](#),
- [ModelName](#), [ModelNumber](#), [ModelUrl](#), [SerialNumber](#),
- [Udn](#), [Upc](#), [PresentationUrl](#), [FileAddress](#),
- [Location](#), [Server](#), [IpAddress](#), [Port](#),
- [Running](#) }

## Public Slots

- void **respondMSearch** (const QString &st, const QString &senderIp, quint16 senderPort)  
*Slot connected to the msearchRequestReceived() signal comming from the ssdp module.*

## Public Member Functions

- **BrisaDevice** (QObject \*parent=0)  
*Creates a BrisaDevice with the given parent QObject.*
- **BrisaDevice** (const BrisaDevice &dev)  
*Copy constructor.*
- **BrisaDevice** (const QString &deviceType, QString friendlyName="", const QString &manufacturer="", const QString &manufacturerURL="", const QString &modelDescription="", const QString &modelName="", const QString &modelNumber="", const QString &modelURL="", const QString &serialNumber="", const QString &UDN="", const QString &UPC="", const QString &presentationURL="", QObject \*parent=0)  
*Creates a BrisaDevice with the given device information.*
- **~BrisaDevice** ()  
*Destructor for BrisaDevice.*
- **BrisaDevice** & **operator=** (const BrisaDevice &dev)  
*Assigns dev to this BrisaDevice and returns a copy.*
- void **setAttribute** (xmlTags key, const QString &value)  
*Attribute setter.*
- QString **getAttribute** (xmlTags key) const  
*Attribute getter.*
- void **addIcon** (const QString &mimetype="", const QString &width="", const QString &height="", const QString &depth="", const QString &url "")  
*Call this method to add a icon to the device.*
- void **addService** (const QString &serviceType="", const QString &serviceId="", const QString &SCPDURL="", const QString &controlURL="", const QString &eventSubURL "")  
*Creates and adds a service to the device with the given information.*
- void **addService** (BrisaService \*serv)  
*Overloads addService().*
- void **addEmbeddedDevice** (const QString &deviceType="", const QString &friendlyName="", const QString &manufacturer="", const QString &manufacturerURL="", const QString &modelDescription="", const QString &modelName="", const QString &modelNumber="", const QString &modelURL="", const QString &serialNumber="", const QString &UDN="", const QString &UPC="", const QString &presentationURL "")  
*Creates and adds a embedded device with the given information to the device.*

- void [addEmbeddedDevice](#) ([BrisaDevice](#) \*newEmbeddedDevice)  
*Overloads [addEmbeddedDevice\(\)](#).*
- QList< [BrisaIcon](#) > [getIconList](#) () const  
*Returns the icon list.*
- QList< [BrisaService](#) \* > [getServiceList](#) () const  
*Returns the service list.*
- QList< [BrisaDevice](#) \* > [getEmbeddedDeviceList](#) () const  
*Returns the embedded device list.*
- [BrisaService](#) \* [getServiceById](#) (const QString &serviceId)  
*Getter for [BrisaService](#) in the service list.*
- [BrisaService](#) \* [getServiceByType](#) (const QString &serviceType)  
*Getter for [BrisaService](#) in the service list.*
- void [clear](#) ()  
*Clears the device information, including services, icons and embedded devices.*
- void [doNotify](#) ()  
*Sends the ssdp:alive messages for root device, embedded devices and services according to the UPnP 1.0 specification.*
- void [doByeBye](#) ()  
*Sends the ssdp:byebye messages for root device, embedded devices and services according to the UPnP 1.0 specification.*
- void [start](#) ()  
*Call this method to join the network and start the device.*
- void [stop](#) ()  
*Stops the device and leaves the network.*

### 6.15.1 Detailed Description

UPnP device implementation. [BrisaDevice](#) provides a easy and fast way to create UPnP devices. Simply create a new [BrisaDevice](#) and call [start\(\)](#) to join the network and be visible to available control points.

To add a service to the device, just create a new [BrisaService](#) with the chosen actions and events and add it to the device by calling [addService\(\)](#). The service will be automatically added to the device and the appropriate webserver urls paths will be created.

Embedded devices are also supported by [BrisaDevice](#). Create a new [BrisaDevice](#) and call [addEmbeddedDevice\(\)](#), the embedded device will be announced when the root device joins the network.

To stop the device and leave the network, simply call the [stop\(\)](#) method, ssdp messages will also be sent for any embedded devices.

See also:

[BrisaUpnp::BrisaService](#)

## 6.15.2 Member Enumeration Documentation

### 6.15.2.1 enum BrisaUpnp::BrisaDevice::xmlTags

Enumerator:

*Major*

*Minor*

*UrlBase*

*DeviceType*

*FriendlyName*

*Manufacturer*

*ManufacturerUrl*

*ModelDescription*

*ModelName*

*ModelNumber*

*ModelUrl*

*SerialNumber*

*Udn*

*Upc*

*PresentationUrl*

*FileAddress*

*Location*

*Server*

*IpAddress*

*Port*

*Running*

## 6.15.3 Constructor & Destructor Documentation

### 6.15.3.1 BrisaDevice::BrisaDevice (QObject \* *parent* = 0)

Creates a [BrisaDevice](#) with the given parent QObject.

Referenced by addEmbeddedDevice().

### 6.15.3.2 BrisaDevice::BrisaDevice (const BrisaDevice & *dev*)

Copy constructor.

---

**6.15.3.3 BrisaDevice::BrisaDevice (const QString & *deviceType*, QString *friendlyName* = "", const QString & *manufacturer* = "", const QString & *manufacturerURL* = "", const QString & *modelDescription* = "", const QString & *modelName* = "", const QString & *modelNumber* = "", const QString & *modelURL* = "", const QString & *serialNumber* = "", const QString & *UDN* = "", const QString & *UPC* = "", const QString & *presentationURL* = "", QObject \* *parent* = 0)**

Creates a [BrisaDevice](#) with the given device information.

**6.15.3.4 BrisaDevice::~BrisaDevice ()**

Destructor for [BrisaDevice](#). Stops the device if running.

## 6.15.4 Member Function Documentation

**6.15.4.1 void BrisaDevice::addEmbeddedDevice (BrisaDevice \* *newEmbeddedDevice*)**

Overloads [addEmbeddedDevice\(\)](#). Create a new [BrisaDevice](#) and call this method to add it as a embedded device to a root device. We recommend using this method for better object orientation.

**6.15.4.2 void BrisaDevice::addEmbeddedDevice (const QString & *deviceType* = "", const QString & *friendlyName* = "", const QString & *manufacturer* = "", const QString & *manufacturerURL* = "", const QString & *modelDescription* = "", const QString & *modelName* = "", const QString & *modelNumber* = "", const QString & *modelURL* = "", const QString & *serialNumber* = "", const QString & *UDN* = "", const QString & *UPC* = "", const QString & *presentationURL* = "")**

Creates and adds a embedded device with the given information to the device.

**6.15.4.3 void BrisaDevice::addIcon (const QString & *mimetype* = "", const QString & *width* = "", const QString & *height* = "", const QString & *depth* = "", const QString & *url* = "")**

Call this method to add a icon to the device.

**6.15.4.4 void BrisaDevice::addService (BrisaService \* *servdev*)**

Overloads [addService\(\)](#). Create a [BrisaService](#) and add it to the device. We recommend using this method for better object orientation.

### See also:

[BrisaUpnp::BrisaService](#)

---

**6.15.4.5 void BrisaDevice::addService (const QString & *serviceType* = "", const QString & *serviceId* = "", const QString & *SCPDURL* = "", const QString & *controlURL* = "", const QString & *eventSubURL* = "")**

Creates and adds a service to the device with the given information.

**6.15.4.6 void BrisaDevice::clear ()**

Clears the device information, including services, icons and embedded devices.

**6.15.4.7 void BrisaDevice::doByeBye ()**

Sends the ssdp:byebye messages for root device, embedded devices and services according to the UPnP 1.0 specification.

Referenced by stop().

**6.15.4.8 void BrisaDevice::doNotify ()**

Sends the ssdp:alive messages for root device, embedded devices and services according to the UPnP 1.0 specification.

Referenced by start().

**6.15.4.9 QString BrisaDevice::getAttribute (xmlTags *key*) const**

Attribute getter.

**See also:**

[setAttribute\(\)](#)

Referenced by BrisaDevice(), operator=(), and BrisaUpnp::BrisaDeviceXMLHandler::xmlGenerator().

**6.15.4.10 QList< BrisaDevice \* > BrisaDevice::getEmbeddedDeviceList () const**

Returns the embedded device list.

**See also:**

[getIconList\(\)](#) , [getServiceList\(\)](#)

Referenced by operator=().

**6.15.4.11 QList< BrisaIcon > BrisaDevice::getIconList () const**

Returns the icon list.

**See also:**

[getEmbeddedDeviceList\(\)](#) , [getServiceList\(\)](#)

Referenced by operator=().

**6.15.4.12 BrisaService \* BrisaDevice::getServiceById (const QString & *serviceId*)**

Getter for [BrisaService](#) in the service list.

**6.15.4.13 BrisaService \* BrisaDevice::getServiceByType (const QString & *serviceType*)**

Getter for [BrisaService](#) in the service list.

**6.15.4.14 QList< BrisaService \* > BrisaDevice::getServiceList () const**

Returns the service list.

**See also:**

[getEmbeddedDeviceList\(\)](#) , [getIconList\(\)](#)

Referenced by operator=().

**6.15.4.15 BrisaDevice & BrisaDevice::operator= (const BrisaDevice & *dev*)**

Assigns dev to this [BrisaDevice](#) and returns a copy.

**6.15.4.16 void BrisaDevice::respondMSearch (const QString & *st*, const QString & *senderIp*, quint16 *senderPort*) [slot]**

Slot connected to the msearchRequestReceived() signal comming from the ssdp module. It parses the search type and responds accordingly.

**See also:**

[respondMSearchAll\(\)](#)

Referenced by [BrisaDevice\(\)](#).

**6.15.4.17 void BrisaDevice::setAttribute (xmlTags *key*, const QString & *v*)**

Attribute setter.

**See also:**

[getAttribute\(\)](#)

**6.15.4.18 void BrisaDevice::start ()**

Call this method to join the network and start the device.

**See also:**

[stop\(\)](#)

**6.15.4.19 void BrisaDevice::stop ()**

Stops the device and leaves the network.

**See also:**

[start\(\)](#)

Referenced by [~BrisaDevice\(\)](#).

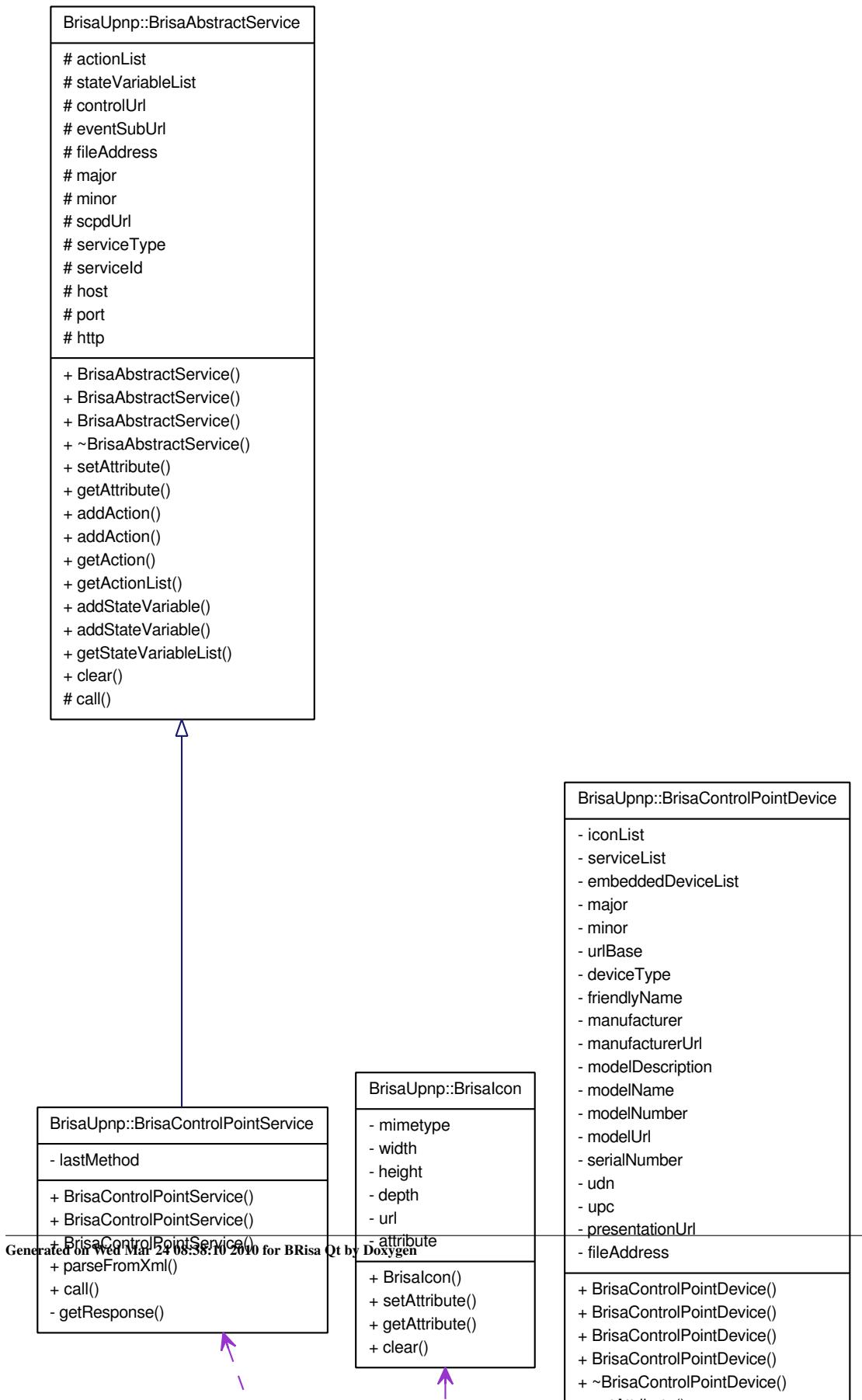
The documentation for this class was generated from the following files:

- [src/upnp/device/brisadevice.h](#)
- [src/upnp/device/brisadevice.cpp](#)



## 6.16 BrisaUpnp::BrisaDeviceParserContext Class Reference

#include <brisadevicexmlhandlercp.h>Collaboration diagram for BrisaUpnp::BrisaDeviceParserContext:



## Public Member Functions

- `BrisaDeviceParserContext (BrisaDeviceParserContext *parent=0, BrisaControlPointDevice *target=0)`
- `BrisaIcon * getIcon (void)`
- `BrisaControlPointDevice * getDevice (void)`
- `BrisaControlPointService * getService (void)`
- `void setIcon (BrisaIcon *icon)`
- `void setDevice (BrisaControlPointDevice *device)`
- `void setService (BrisaControlPointService *service)`
- `bool hasParent (void)`
- `BrisaDeviceParserContext * getParent (void)`

## Public Attributes

- `int stateSkip`
- `SaxParserState state`

### 6.16.1 Constructor & Destructor Documentation

**6.16.1.1 BrisaUpnp::BrisaDeviceParserContext::BrisaDeviceParserContext**  
`(BrisaDeviceParserContext *parent = 0, BrisaControlPointDevice *target = 0)`  
`[inline]`

### 6.16.2 Member Function Documentation

**6.16.2.1 BrisaControlPointDevice\* BrisaUpnp::BrisaDeviceParserContext::getDevice (void)**  
`[inline]`

Referenced by BrisaUpnp::BrisaDeviceXMLHandlerCP::characters(), BrisaUpnp::BrisaDeviceXMLHandlerCP::endElement(), and BrisaUpnp::BrisaDeviceXMLHandlerCP::startElement().

**6.16.2.2 BrisaIcon\* BrisaUpnp::BrisaDeviceParserContext::getIcon (void) [inline]**

Referenced by BrisaUpnp::BrisaDeviceXMLHandlerCP::characters(), BrisaUpnp::BrisaDeviceXMLHandlerCP::endElement(), and BrisaUpnp::BrisaDeviceXMLHandlerCP::startElement().

**6.16.2.3 BrisaDeviceParserContext\* BrisaUpnp::BrisaDeviceParserContext::getParent (void)**  
`[inline]`

Referenced by BrisaUpnp::BrisaDeviceXMLHandlerCP::endElement().

**6.16.2.4 BrisaControlPointService\* BrisaUpnp::BrisaDeviceParserContext::getService (void)**  
`[inline]`

Referenced by BrisaUpnp::BrisaDeviceXMLHandlerCP::characters(), BrisaUpnp::BrisaDeviceXMLHandlerCP::endElement(), and BrisaUpnp::BrisaDeviceXMLHandlerCP::startElement().

**6.16.2.5 bool BrisaUpnp::BrisaDeviceParserContext::hasParent (void) [inline]**

Referenced by BrisaUpnp::BrisaDeviceXMLHandlerCP::EndElement().

**6.16.2.6 void BrisaUpnp::BrisaDeviceParserContext::setDevice (BrisaControlPointDevice \* device) [inline]**

Referenced by BrisaUpnp::BrisaDeviceXMLHandlerCP::StartElement().

**6.16.2.7 void BrisaUpnp::BrisaDeviceParserContext::setIcon (BrisaIcon \* icon) [inline]**

Referenced by BrisaUpnp::BrisaDeviceXMLHandlerCP::EndElement(), and BrisaUpnp::BrisaDeviceXMLHandlerCP::StartElement().

**6.16.2.8 void BrisaUpnp::BrisaDeviceParserContext::setService (BrisaControlPointService \* service) [inline]**

Referenced by BrisaUpnp::BrisaDeviceXMLHandlerCP::EndElement(), and BrisaUpnp::BrisaDeviceXMLHandlerCP::StartElement().

## 6.16.3 Member Data Documentation

**6.16.3.1 SaxParserState BrisaUpnp::BrisaDeviceParserContext::state**

Referenced by BrisaUpnp::BrisaDeviceXMLHandlerCP::characters(), BrisaUpnp::BrisaDeviceXMLHandlerCP::EndElement(), BrisaUpnp::BrisaDeviceXMLHandlerCP::parseDevice(), and BrisaUpnp::BrisaDeviceXMLHandlerCP::StartElement().

**6.16.3.2 int BrisaUpnp::BrisaDeviceParserContext::stateSkip**

Referenced by BrisaUpnp::BrisaDeviceXMLHandlerCP::EndElement(), BrisaUpnp::BrisaDeviceXMLHandlerCP::parseDevice(), and BrisaUpnp::BrisaDeviceXMLHandlerCP::StartElement().

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

- src/upnp/controlpoint/[brisadevicexmlhandlercp.h](#)

## 6.17 BrisaUpnp::BrisaDeviceXMLHandler Class Reference

```
#include <brisadevicexmlhandler.h>
```

### Public Member Functions

- void [xmlGenerator \(BrisaDevice \\*device, QFile \\*file\)](#)

#### 6.17.1 Member Function Documentation

##### 6.17.1.1 void BrisaDeviceXMLHandler::xmlGenerator (BrisaDevice \* *device*, QFile \* *file*)

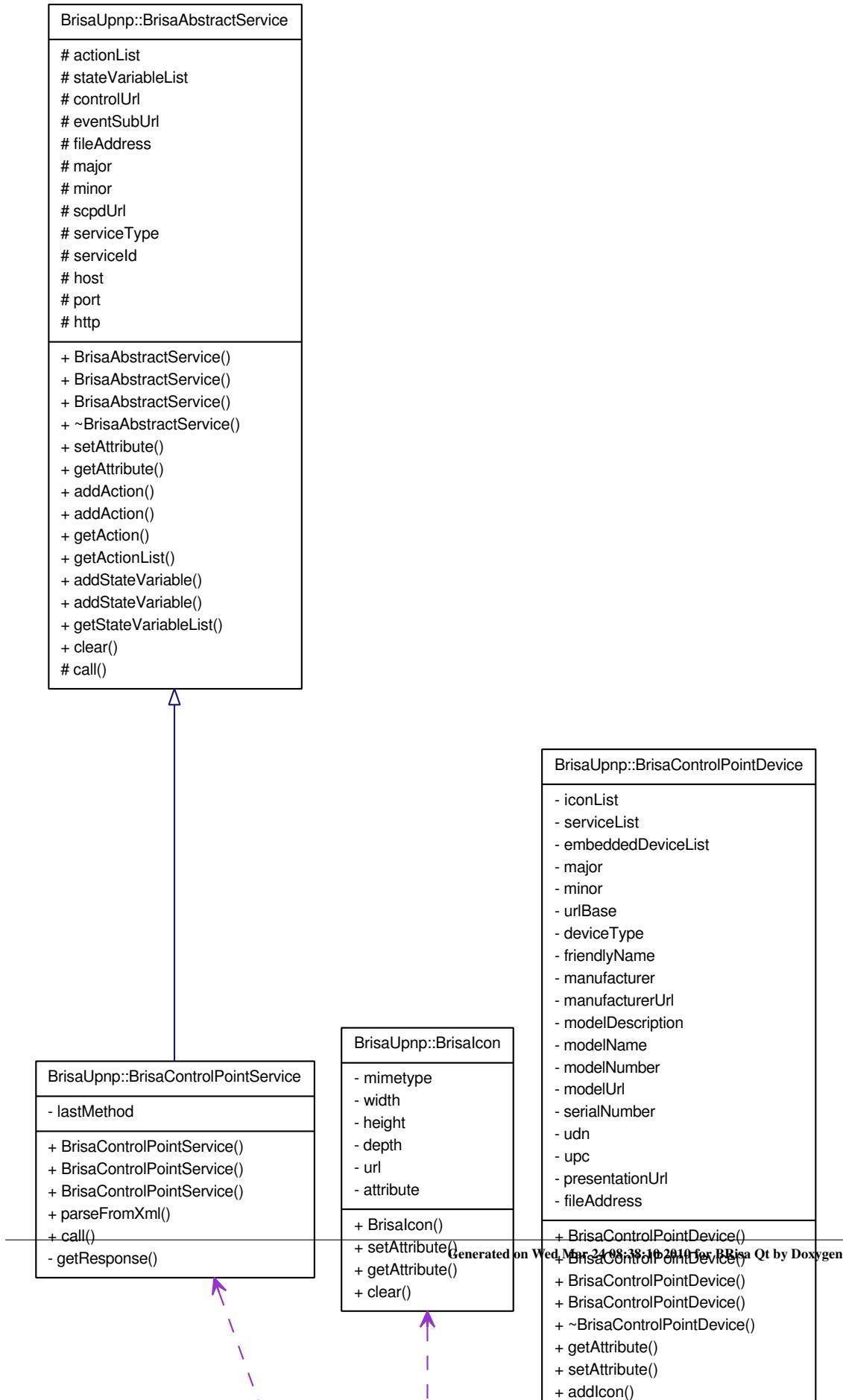
The documentation for this class was generated from the following files:

- [src/upnp/device/brisadevicexmlhandler.h](#)
- [src/upnp/device/brisadevicexmlhandler.cpp](#)

## 6.18 BrisaUpnp::BrisaDeviceXMLHandlerCP Class Reference

[BrisaDeviceXMLHandlerCP](#) creates a device from a xml description file, with all it's attributes, it let's it ready to be used.

#include <BrisaUpnp/BrisaDeviceXMLHandlerCP>Collaboration diagram for BrisaUpnp::BrisaDeviceXMLHandlerCP:



## Public Member Functions

- void [parseDevice \(BrisaControlPointDevice \\*device, QTemporaryFile \\*tmp\)](#)

*Method that initializes device attributes from tmp file.*

## Protected Member Functions

- bool [startElement \(const QString &namespaceURI, const QString &localName, const QString &qName, const QXmlAttributes &attributes\)](#)

*Method inherited from QXmlDefaultHandler it's called in every beginning tag, depending on tag the context state passes to a different state so that it can process separately.*

- bool [endElement \(const QString &namespaceURI, const QString &localName, const QString &qName\)](#)

*Method that is called in the tag end, this method add the attribute to the device so that attributes can be initialized in device.*

- bool [characters \(const QString &str\)](#)

*Method that set properly the attribute value to value that it is the content between the beginning tag and the finish one.*

### 6.18.1 Detailed Description

[BrisaDeviceXMLHandlerCP](#) creates a device from a xml description file, with all it's attributes, it let's it ready to be used.

### 6.18.2 Member Function Documentation

#### 6.18.2.1 bool BrisaDeviceXMLHandlerCP::characters (const QString & str) [protected]

Method that set properly the attribute value to value that it is the content between the beginning tag and the finish one. It's important to say that in each state it performs a different action.

#### 6.18.2.2 bool BrisaDeviceXMLHandlerCP::endElement (const QString & namespaceURI, const QString & localName, const QString & qName) [protected]

Method that is called in the tag end, this method add the attribute to the device so that attributes can be initialized in device. It's important to say that in each state it performs a different action.

#### 6.18.2.3 void BrisaDeviceXMLHandlerCP::parseDevice (BrisaControlPointDevice \* device, QTemporaryFile \* tmp)

Method that initializes *device* attributes from *tmp* file.

#### 6.18.2.4 **bool BrisaDeviceXMLHandlerCP::startElement (const QString & *namespaceURI*, const QString & *localName*, const QString & *qName*, const QXmlAttributes & *attributes*) [protected]**

Method inherited from QXmlDefaultHandler it's called in every beginning tag, depending on tag the context state passes to a different state so that it can process separately. It's important to say that in each state it performs a different action.

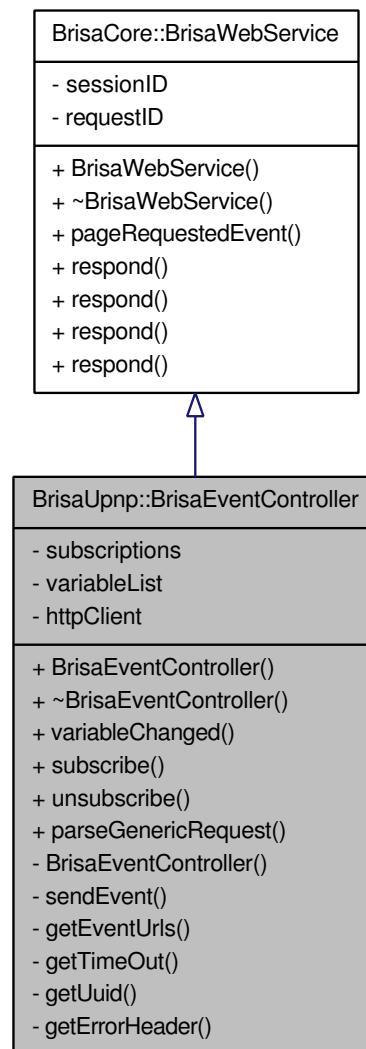
The documentation for this class was generated from the following files:

- src/upnp/controlpoint/[brisadevicexmlhandlercp.h](#)
- src/upnp/controlpoint/[brisadevicexmlhandlercp.cpp](#)

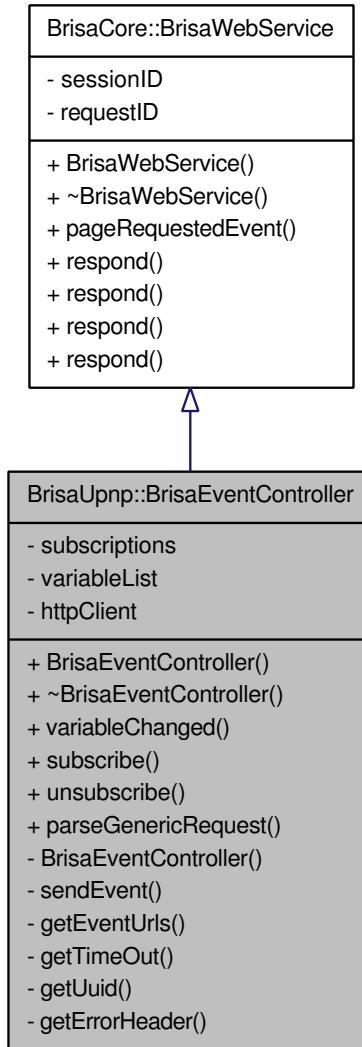
## 6.19 BrisaUpnp::BrisaEventController Class Reference

```
#include <brisaeventcontroller.h>Inheritance  
BrisaUpnp::BrisaEventController:
```

diagram for



Collaboration diagram for BrisaUpnp::BrisaEventController:



## Public Slots

- void **variableChanged** (*BrisaStateVariable* \*variable)
 

*Slot that shall be called when some service's state variable change.*
- void **subscribe** (const QMultiHash< QString, QString > &subscriberInfo, int sessionId, int requestId)
 

*Creates a subscription for the given subscriberInfo, sessionId and requestId.*
- void **unsubscribe** (const QMultiHash< QString, QString > &subscriberInfo, int sessionId, int requestId)
 

*Removes the subscription for the given subscriberInfo, sessionId and requestId.*
- void **parseGenericRequest** (const QString &method, const QMultiHash< QString, QString > &headers, const QByteArray &requestContent, int sessionId, int requestId)

Parses a generic request to web service and calls the local related methods as necessary.

## Public Member Functions

- **BrisaEventController** (QxtAbstractWebSessionManager \**sessionManager*, QList< [BrisaStateVariable](#) \*> \**stateVariableList*, QObject \**parent*=0)  
*Constructs a service's event controller with the given sessionManager, service's stateVariableList and parent.*
- **~BrisaEventController** ()  
*Destructor.*

### 6.19.1 Constructor & Destructor Documentation

#### 6.19.1.1 **BrisaEventController::BrisaEventController** (QxtAbstractWebSessionManager \**sessionManager*, QList< [BrisaStateVariable](#) \*> \**stateVariableList*, QObject \**parent* = 0)

Constructs a service's event controller with the given *sessionManager*, service's *stateVariableList* and *parent*.

#### 6.19.1.2 **BrisaEventController::~BrisaEventController** ()

*Destructor.*

## 6.19.2 Member Function Documentation

#### 6.19.2.1 **void BrisaEventController::parseGenericRequest** (const QString & *method*, const QMultiHash< QString, QString > & *headers*, const QByteArray & *requestContent*, int *sessionId*, int *requestId* [slot])

Parses a generic request to web service and calls the local related methods as necessary.

Referenced by [BrisaEventController\(\)](#).

#### 6.19.2.2 **void BrisaEventController::subscribe** (const QMultiHash< QString, QString > & *subscriberInfo*, int *sessionId*, int *requestId*) [slot]

Creates a subscription for the given *subscriberInfo*, *sessionId* and *requestId*.

Referenced by [parseGenericRequest\(\)](#).

#### 6.19.2.3 **void BrisaEventController::unsubscribe** (const QMultiHash< QString, QString > & *subscriberInfo*, int *sessionId*, int *requestId*) [slot]

Removes the subscription for the given *subscriberInfo*, *sessionId* and *requestId*.

Referenced by [parseGenericRequest\(\)](#).

**6.19.2.4 void BrisaEventController::variableChanged (BrisaStateVariable \* *variable*) [slot]**

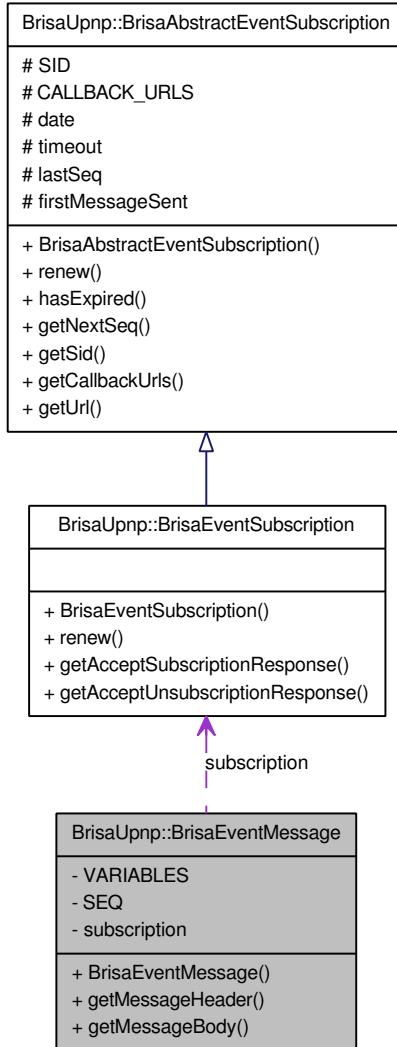
Slot that shall be called when some service's state *variable* change.

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

- src/upnp/device/[brisaeventcontroller.h](#)
- src/upnp/device/[brisaeventcontroller.cpp](#)

## 6.20 BrisaUpnp::BrisaEventMessage Class Reference

#include <brisaeventmessage.h>Collaboration diagram for BrisaUpnp::BrisaEventMessage:



### Public Member Functions

- **BrisaEventMessage** ([BrisaEventSubscription](#) &subscription, const [QList< BrisaStateVariable \\* >](#) \*variables, [QObject](#) \*parent=0)
- Constructs a new event message to the given subscription and related to the given variables, with the given parent object.*
- [QHttpRequestHeader getMessageHeader](#) () const
- Returns this event message's http header.*
- [QByteArray getMessageBody](#) () const
- Returns this event message's http body.*

### 6.20.1 Constructor & Destructor Documentation

#### 6.20.1.1 BrisaEventMessage::BrisaEventMessage (BrisaEventSubscription & *subscription*, const QList< BrisaStateVariable \* > \* *variables*, QObject \* *parent* = 0)

Constructs a new event message to the given *subscription* and related to the given *variables*, with the given *parent* object.

### 6.20.2 Member Function Documentation

#### 6.20.2.1 QByteArray BrisaEventMessage::getMessageBody () const

Returns this event message's http body.

Referenced by getMessageHeader().

#### 6.20.2.2 QHttpRequestHeader BrisaEventMessage::getMessageHeader () const

Returns this event message's http header.

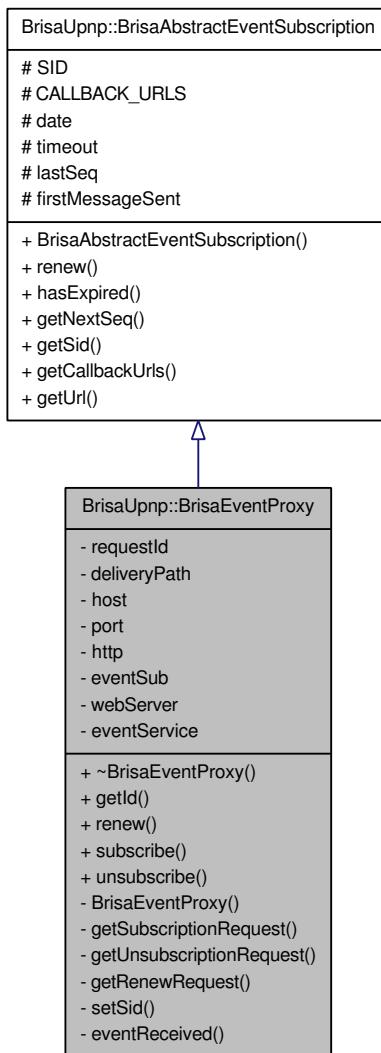
The documentation for this class was generated from the following files:

- src/upnp/device/[brisaeventmessage.h](#)
- src/upnp/device/[brisaeventmessage.cpp](#)

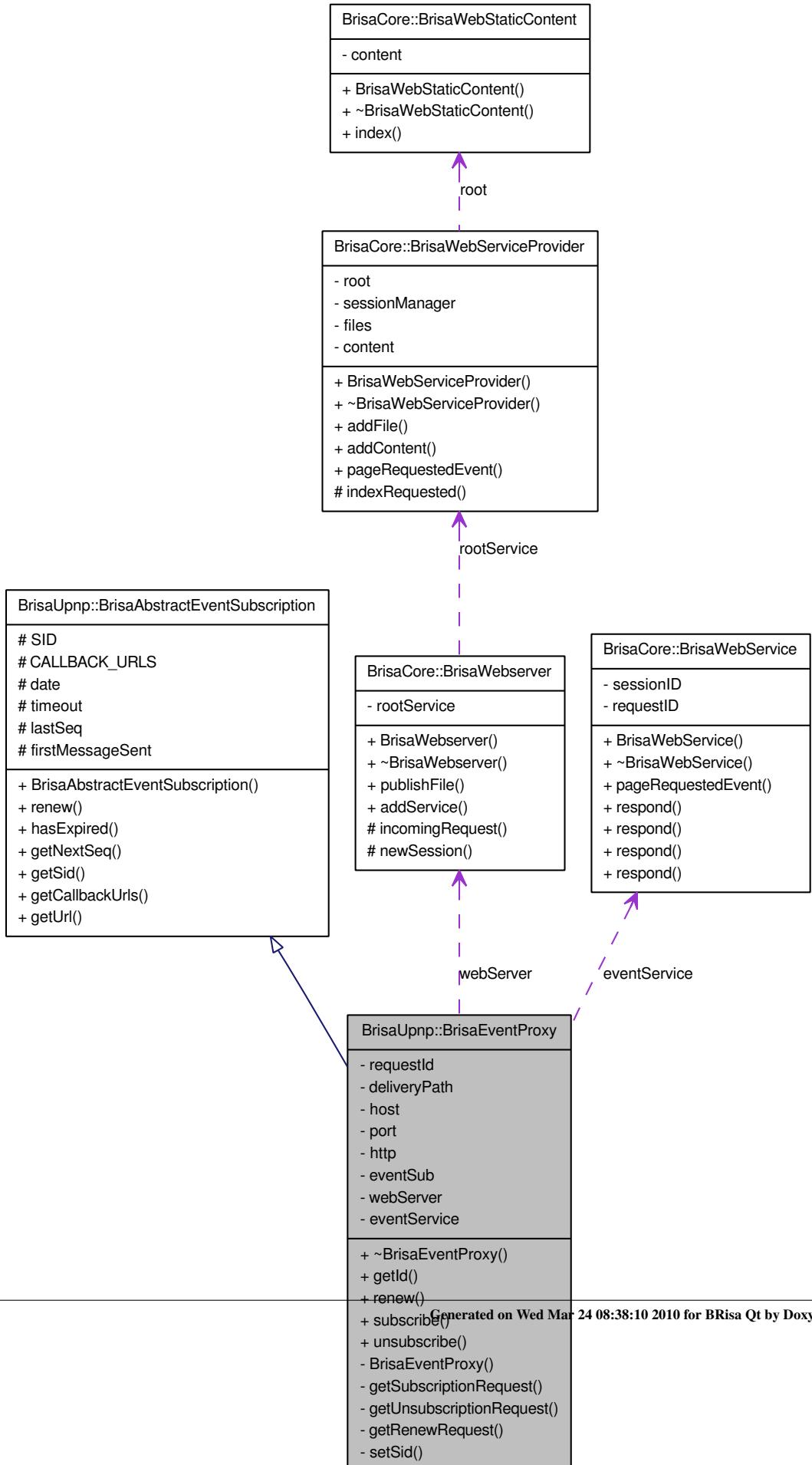
## 6.21 BrisaUpnp::BrisaEventProxy Class Reference

Class that implements the event part in control point side in Brisa Qt, this class makes the operations of subscribe, renew subscription and unsubscribe.

#include <BrisaUpnp/BrisaEventProxy> Inheritance diagram for BrisaUpnp::BrisaEventProxy:



Collaboration diagram for BrisaUpnp::BrisaEventProxy:



## Signals

- void `eventNotification (BrisaEventProxy *subscription, QMap< QString, QString > eventingVariables)`

## Public Member Functions

- `~BrisaEventProxy ()`

*Destructor.*

- int `getId ()`

*Gets the request id.*

- void `renew (const int &newTimeout=-1)`

*Renew the subscribe in a event for the newTimeout passed.*

- void `subscribe (const int timeout=-1)`

*Subscribe for the events from a service subscriptions will last the timeout passed.*

- void `unsubscribe ()`

*Unsubscribe the events from a service, using this the user won't receive more event responses.*

## Friends

- class `BrisaControlPoint`

### 6.21.1 Detailed Description

Class that implements the event part in control point side in Brisa Qt, this class makes the operations of subscribe, renew subscription and unsubscribe.

### 6.21.2 Constructor & Destructor Documentation

#### 6.21.2.1 BrisaEventProxy::~BrisaEventProxy ()

*Destructor.*

### 6.21.3 Member Function Documentation

#### 6.21.3.1 void BrisaUpnp::BrisaEventProxy::eventNotification (BrisaEventProxy \* *subscription*, QMap< QString, QString > *eventingVariables*) [signal]

#### 6.21.3.2 int BrisaEventProxy::getId ()

*Gets the request id.*

**6.21.3.3 void BrisaEventProxy::renew (const int & newTimeout = -1) [virtual]**

Renew the subscribe in a event for the *newTimeout* passed.

Implements [BrisaUpnp::BrisaAbstractEventSubscription](#).

**6.21.3.4 void BrisaEventProxy::subscribe (const int timeout = -1)**

Subscribe for the events from a service subscriptions will last the *timeout* passed.

**6.21.3.5 void BrisaEventProxy::unsubscribe (void)**

Unsubscribe the events from a service, using this the user won't receive more event responses.

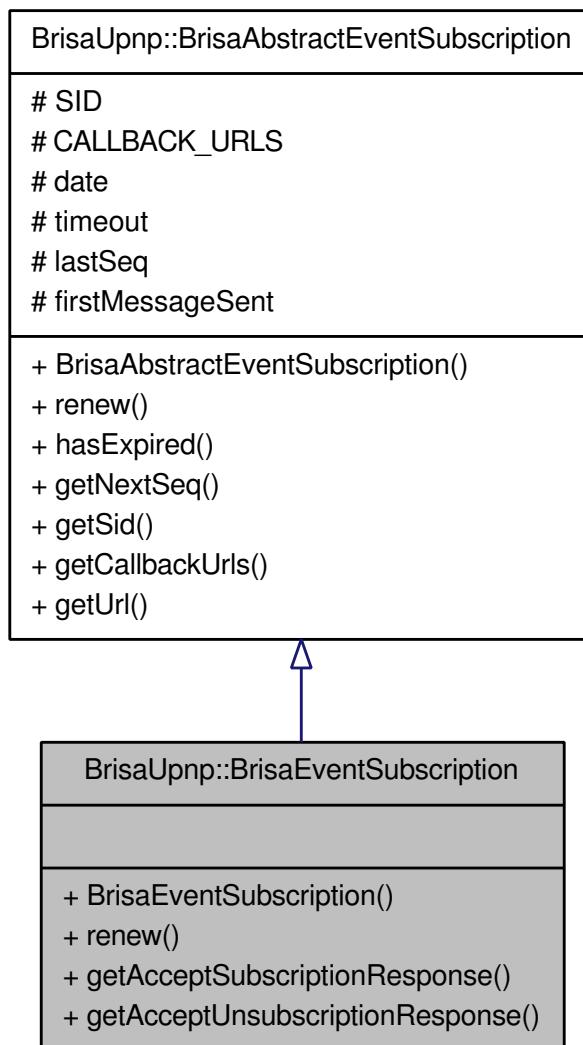
**6.21.4 Friends And Related Function Documentation****6.21.4.1 friend class BrisaControlPoint [friend]**

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

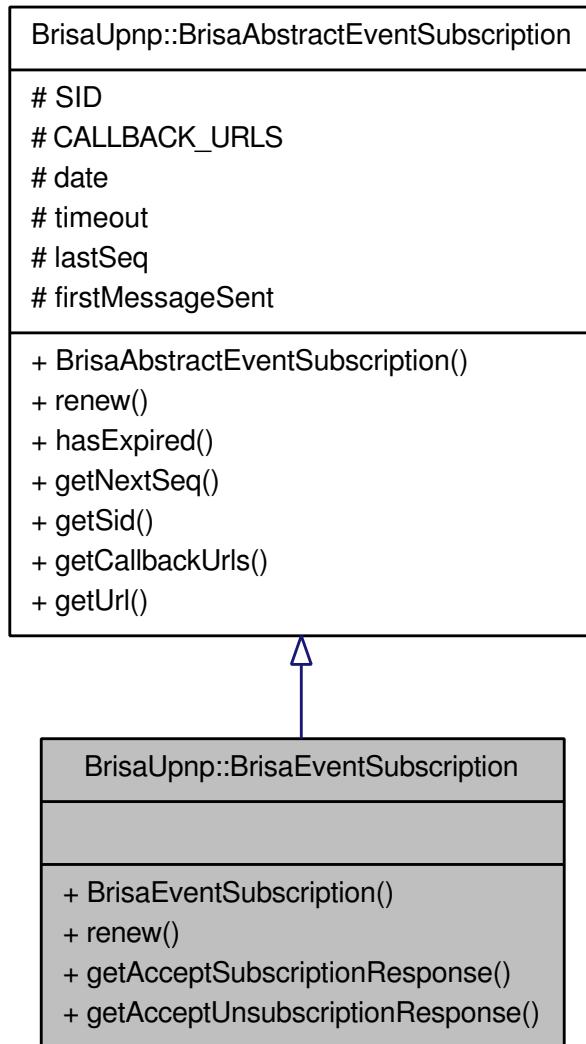
- src/upnp/controlpoint/[brisaeventproxy.h](#)
- src/upnp/controlpoint/[brisaeventproxy.cpp](#)

## 6.22 BrisaUpnp::BrisaEventSubscription Class Reference

#include <brisaeventsubscription.h> Inheritance diagram for BrisaUpnp::BrisaEventSubscription:



Collaboration diagram for BrisaUpnp::BrisaEventSubscription:



## Public Member Functions

- `BrisaEventSubscription (const QString &sid, const QStringList &callbackUrls, const int &timeout=-1, QObject *parent=0)`
- `void renew (const int &newTimeout=-1)`

*Renews the subscription for the given newTimeout.*

- `QHttpResponseHeader getAcceptSubscriptionResponse () const`
- `QHttpResponseHeader getAcceptUnsubscriptionResponse () const`

### 6.22.1 Constructor & Destructor Documentation

**6.22.1.1 BrisaUpnp::BrisaEventSubscription::BrisaEventSubscription (const QString & *sid*, const QStringList & *callbackUrls*, const int & *timeout* = -1, QObject \* *parent* = 0) [inline]**

### 6.22.2 Member Function Documentation

**6.22.2.1 QHttpResponseHeader BrisaEventSubscription::getAcceptSubscriptionResponse () const**

Referenced by BrisaUpnp::BrisaEventController::subscribe().

**6.22.2.2 QHttpResponseHeader BrisaEventSubscription::getAcceptUnsubscriptionResponse () const**

**6.22.2.3 void BrisaEventSubscription::renew (const int & *newTimeout* = -1) [virtual]**

Renews the subscription for the given *newTimeout*.

Implements [BrisaUpnp::BrisaAbstractEventSubscription](#).

Referenced by BrisaUpnp::BrisaEventController::subscribe().

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

- [src/upnp/device/brisaeventsubscription.h](#)
- [src/upnp/device/brisaeventsubscription.cpp](#)

## 6.23 BrisaUpnp::BrisaIcon Class Reference

```
#include <brisaicon.h>
```

### Public Types

- enum `xmlIconTags` {
 `Mimetype`, `Width`, `Height`, `Depth`,  
`Url` }

### Public Member Functions

- `BrisaIcon` (QString `mimetype`= "", QString `width`= "", QString `height`= "", QString `depth`= "", QString `url`= "")
- void `setAttribute` (`xmlIconTags` `key`, QString `v`)
- QString `getAttribute` (`xmlIconTags` `key`)
- void `clear` ()

#### 6.23.1 Member Enumeration Documentation

##### 6.23.1.1 enum BrisaUpnp::BrisaIcon::xmlIconTags

Enumerator:

*Mimetype*  
*Width*  
*Height*  
*Depth*  
*Url*

#### 6.23.2 Constructor & Destructor Documentation

##### 6.23.2.1 BrisaIcon::BrisaIcon (QString `mimetype` = " ", QString `width` = " ", QString `height` = " ", QString `depth` = " ", QString `url` = " ")

#### 6.23.3 Member Function Documentation

##### 6.23.3.1 void BrisaIcon::clear ()

##### 6.23.3.2 QString BrisaIcon::getAttribute (xmlIconTags `key`)

##### 6.23.3.3 void BrisaIcon::setAttribute (xmlIconTags `key`, QString `v`)

Referenced by BrisaUpnp::BrisaDeviceXMLHandlerCP::characters().

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

- src/upnp/[brisaiicon.h](#)
- src/upnp/[brisaiicon.cpp](#)

## 6.24 BrisaUpnp::BrisaMSearchClientCP Class Reference

SSDP MSearch implementation for UPnP control points.

```
#include <BrisaUpnp/BrisaMSearchClientCP>
```

### Public Slots

- void [discover\(\)](#)

### Signals

- void [msearchResponseReceived\(const QString &usn, const QString &location, const QString &st, const QString &ext, const QString &server, const QString &cacheControl\)](#)

### Public Member Functions

- [BrisaMSearchClientCP\(QObject \\*parent=0, const QString &type=DEFAULT\\_SEARCH\\_TYPE, int mx=5\)](#)
- virtual ~[BrisaMSearchClientCP\(\)](#)
- void [doubleDiscover\(\)](#)
- bool [isRunning\(\)](#) const
- void [start\(int interval=DEFAULT\\_SEARCH\\_TIME\)](#)
- void [stop\(\)](#)

#### 6.24.1 Detailed Description

SSDP MSearch implementation for UPnP control points. Create a new [BrisaMSearchClientCP](#) with the desired service type and mx values, and call [start\(\)](#) to begin sending discovery messages to possible devices in the multicast group. When a device responds to a msearch request, [msearchResponseReceived\(\)](#) signal is emitted.

#### 6.24.2 Constructor & Destructor Documentation

**6.24.2.1 BrisaMSearchClientCP::BrisaMSearchClientCP (QObject \**parent* = 0, const QString &*type* = DEFAULT\_SEARCH\_TYPE, int *mx* = 5)**

**6.24.2.2 BrisaMSearchClientCP::~BrisaMSearchClientCP () [virtual]**

#### 6.24.3 Member Function Documentation

**6.24.3.1 void BrisaMSearchClientCP::discover () [slot]**

Referenced by [BrisaMSearchClientCP\(\)](#), [BrisaUpnp::BrisaControlPoint::discover\(\)](#), and [doubleDiscover\(\)](#).

**6.24.3.2 void BrisaMSearchClientCP::doubleDiscover ()****6.24.3.3 bool BrisaMSearchClientCP::isRunning () const**

Referenced by start(), stop(), and ~BrisaMSearchClientCP().

**6.24.3.4 void BrisaUpnp::BrisaMSearchClientCP::msearchResponseReceived (const QString & *usn*, const QString & *location*, const QString & *st*, const QString & *ext*, const QString & *server*, const QString & *cacheControl*) [signal]****6.24.3.5 void BrisaMSearchClientCP::start (int *interval* = DEFAULT\_SEARCH\_TIME)**

Referenced by BrisaUpnp::BrisaControlPoint::start().

**6.24.3.6 void BrisaMSearchClientCP::stop ()**

Referenced by BrisaUpnp::BrisaControlPoint::stop(), and ~BrisaMSearchClientCP().

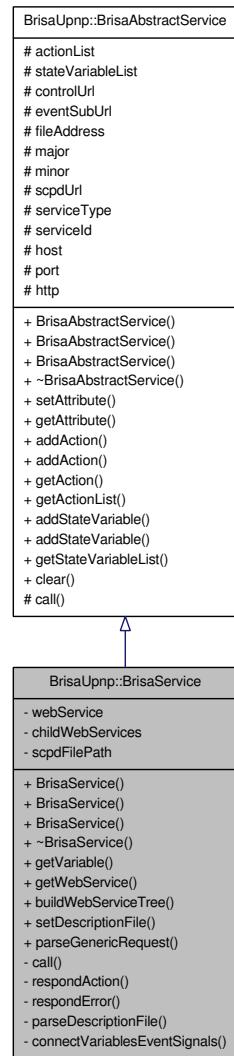
The documentation for this class was generated from the following files:

- src/upnp/controlpoint/[brisamsearchclientcp.h](#)
- src/upnp/controlpoint/[brisamsearchclientcp.cpp](#)

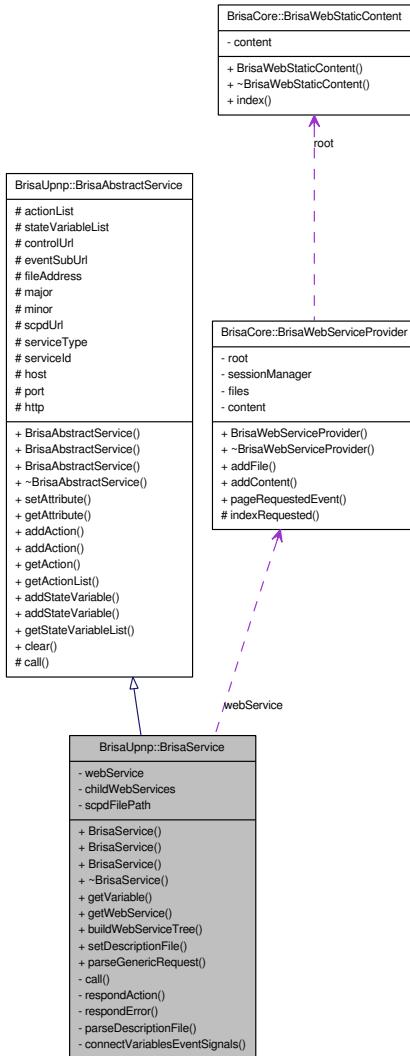
## 6.25 BrisaUpnp::BrisaService Class Reference

UPnP service abstraction.

#include <BrisaUpnp/BrisaService> Inheritance diagram for BrisaUpnp::BrisaService:



Collaboration diagram for BrisaUpnp::BrisaService:



## Public Slots

- void `parseGenericRequest` (const QString &method, const QMultiHash<QString, QString> &headers, const QByteArray &requestContent, int sessionId, int requestId)
- Parses the genericRequestReceived() signal coming from the webservice.*

## Public Member Functions

- `BrisaService` (QObject \*parent=0)  
*Constructs a BrisaService with the given parent.*
- `BrisaService` (const QString &`serviceType`, const QString &`serviceId`='', const QString &`scpdUrl`='', const QString &`controlUrl`='', const QString &`eventSubUrl`='', const QString &`host`='', QObject \*parent=0)

*Constructs a [BrisaService](#) with the given service information.*

- [BrisaService \(BrisaService &service\)](#)

*Copy constructor.*

- [~BrisaService \(\)](#)

*Destructor.*

- [BrisaStateVariable \\* getVariable \(const QString &variableName\)](#)

*Returns the requested [BrisaStateVariable](#), if the variable doesn't exists it return 0.*

- [BrisaWebServiceProvider \\* getWebService \(\)](#)

*Returns the web service.*

- [void buildWebServiceTree \(QxtAbstractWebSessionManager \\*sessionManager\)](#)

*This method creates all the webservice related stuff.*

- [void setDescriptionFile \(const QString &scpdFilePath\)](#)

*Sets the service file path.*

## 6.25.1 Detailed Description

UPnP service abstraction. [BrisaService](#) provides a convinient way to create UPnP services. [BrisaService](#) itself contains a webservice, so integration with the webserver is simple. You can simply create your service actions with [BrisaAction](#), then create a new [BrisaService](#) with the proper information and simply call [addAction\(\)](#) with the previously created actions. The next step is to add your service to the device.

## 6.25.2 Constructor & Destructor Documentation

### 6.25.2.1 BrisaService::BrisaService (*QObject \*parent = 0*)

Constructs a [BrisaService](#) with the given parent.

### 6.25.2.2 BrisaService::BrisaService (*const QString & serviceType, const QString & serviceId = "", const QString & scpdUrl = "", const QString & controlUrl = "", const QString & eventSubUrl = "", const QString & host = "", QObject \*parent = 0*)

Constructs a [BrisaService](#) with the given service information.

### 6.25.2.3 BrisaService::BrisaService (*BrisaService & service*)

Copy constructor.

### 6.25.2.4 BrisaService::~BrisaService ()

Destructor.

### 6.25.3 Member Function Documentation

#### 6.25.3.1 void BrisaService::buildWebServiceTree (QxtAbstractWebSessionManager \* *sessionManager*)

This method creates all the webservice related stuff. It creates a URL for the control path, another url for the event path, and publishes the service description XML file.

#### 6.25.3.2 BrisaStateVariable \* BrisaService::getVariable (const QString & *variableName*)

Returns the requested [BrisaStateVariable](#), if the variable doesn't exists it return 0.

Referenced by BrisaUpnp::BrisaAction::getStateVariable().

#### 6.25.3.3 BrisaWebServiceProvider \* BrisaService::getWebService ()

Returns the web service.

#### 6.25.3.4 void BrisaService::parseGenericRequest (const QString & *method*, const QMultiHash<QString, QString> & *headers*, const QByteArray & *requestContent*, int *sessionId*, int *requestId*) [slot]

Parses the genericRequestReceived() signal coming from the webservice.

Referenced by buildWebServiceTree().

#### 6.25.3.5 void BrisaService::setDescriptionFile (const QString & *scpdFilePath*)

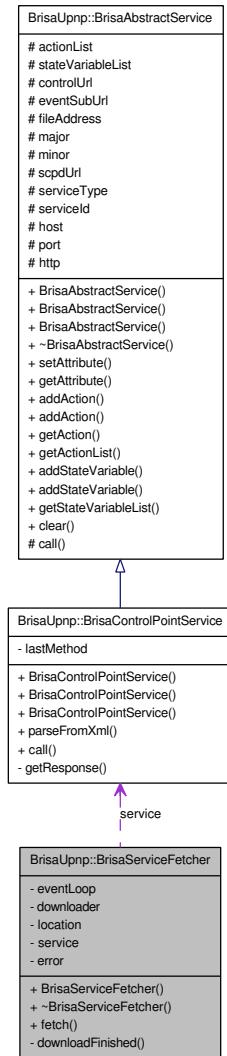
Sets the service file path.

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

- src/upnp/device/[brisaservice.h](#)
- src/upnp/device/[brisaservice.cpp](#)

## 6.26 BrisaUpnp::BrisaServiceFetcher Class Reference

#include <brisadevicexmlhandlercp.h> Collaboration diagram for BrisaUpnp::BrisaServiceFetcher:



## Signals

- void [fetchFinished](#) (void)

## Public Member Functions

- [BrisaServiceFetcher](#) ([BrisaControlPointService](#) \*service, QString location, QObject \*parent=0)
- [~BrisaServiceFetcher](#) ()
- bool [fetch](#) (void)

### 6.26.1 Constructor & Destructor Documentation

**6.26.1.1 BrisaUpnp::BrisaServiceFetcher::BrisaServiceFetcher (BrisaControlPointService \*  
service, QString *location*, QObject \**parent* = 0) [inline]**

**6.26.1.2 BrisaUpnp::BrisaServiceFetcher::~BrisaServiceFetcher () [inline]**

### 6.26.2 Member Function Documentation

**6.26.2.1 bool BrisaUpnp::BrisaServiceFetcher::fetch (void) [inline]**

Referenced by BrisaUpnp::BrisaDeviceXMLHandlerCP::EndElement().

**6.26.2.2 void BrisaUpnp::BrisaServiceFetcher::fetchFinished (void) [signal]**

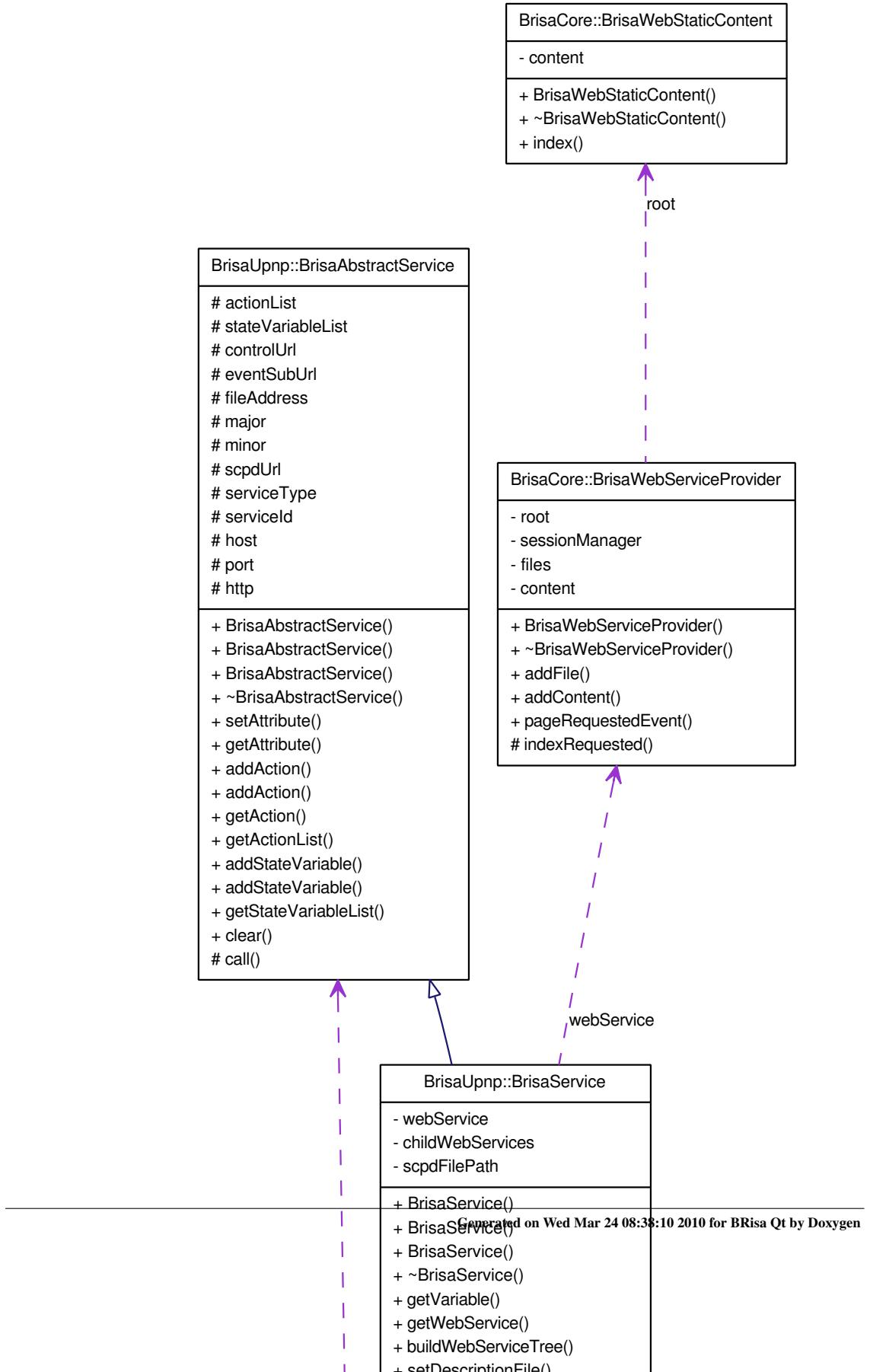
The documentation for this class was generated from the following file:

- src/upnp/controlpoint/[brisadevicexmlhandlercp.h](#)



## 6.27 BrisaUpnp::BrisaServiceParserContext Class Reference

#include <brisaservicexmlhandler.h>Collaboration  
BrisaUpnp::BrisaServiceParserContext:



## Public Member Functions

- `BrisaServiceParserContext (BrisaServiceParserContext *parent=0, BrisaAbstractService *target=0)`
- `BrisaAction * getAction (void)`
- `BrisaAbstractService * getService (void)`
- `BrisaStateVariable * getStateVariable (void)`
- `BrisaArgument * getArguments (void)`
- `void setAction (BrisaAction *action)`
- `void setService (BrisaAbstractService *service)`
- `void setStateVariable (BrisaStateVariable *stateVariable)`
- `void setArguments (BrisaArgument *argument)`
- `bool hasParent (void)`
- `BrisaServiceParserContext * getParent (void)`

## Public Attributes

- `int stateSkip`
- `saxParserState state`

### 6.27.1 Constructor & Destructor Documentation

**6.27.1.1 BrisaUpnp::BrisaServiceParserContext::BrisaServiceParserContext**  
`(BrisaServiceParserContext * parent = 0, BrisaAbstractService * target = 0) [inline]`

### 6.27.2 Member Function Documentation

**6.27.2.1 BrisaAction\* BrisaUpnp::BrisaServiceParserContext::getAction (void) [inline]**

Referenced by `BrisaUpnp::BrisaServiceXMLHandler::characters()`, `BrisaUpnp::BrisaServiceXMLHandler::endElement()`, and `BrisaUpnp::BrisaServiceXMLHandler::startElement()`.

**6.27.2.2 BrisaArgument\* BrisaUpnp::BrisaServiceParserContext::getArguments (void) [inline]**

Referenced by `BrisaUpnp::BrisaServiceXMLHandler::characters()`, `BrisaUpnp::BrisaServiceXMLHandler::endElement()`, and `BrisaUpnp::BrisaServiceXMLHandler::startElement()`.

**6.27.2.3 BrisaServiceParserContext\* BrisaUpnp::BrisaServiceParserContext::getParent (void) [inline]**

**6.27.2.4 BrisaAbstractService\* BrisaUpnp::BrisaServiceParserContext::getService (void) [inline]**

Referenced by `BrisaUpnp::BrisaServiceXMLHandler::characters()`, and `BrisaUpnp::BrisaServiceXMLHandler::endElement()`.

---

**6.27.2.5 BrisaStateVariable\* BrisaUpnp::BrisaServiceParserContext::getStateVariable (void) [inline]**

Referenced by BrisaUpnp::BrisaServiceXMLHandler::characters(), BrisaUpnp::BrisaServiceXMLHandler::endElement(), and BrisaUpnp::BrisaServiceXMLHandler::startElement().

**6.27.2.6 bool BrisaUpnp::BrisaServiceParserContext::hasParent (void) [inline]**
**6.27.2.7 void BrisaUpnp::BrisaServiceParserContext::setAction (BrisaAction \* *action*) [inline]**

Referenced by BrisaUpnp::BrisaServiceXMLHandler::endElement(), and BrisaUpnp::BrisaServiceXMLHandler::startElement().

**6.27.2.8 void BrisaUpnp::BrisaServiceParserContext::setArgument (BrisaArgument \* *argument*) [inline]**

Referenced by BrisaUpnp::BrisaServiceXMLHandler::endElement(), and BrisaUpnp::BrisaServiceXMLHandler::startElement().

**6.27.2.9 void BrisaUpnp::BrisaServiceParserContext::setService (BrisaAbstractService \* *service*) [inline]**
**6.27.2.10 void BrisaUpnp::BrisaServiceParserContext::setStateVariable (BrisaStateVariable \* *stateVariable*) [inline]**

Referenced by BrisaUpnp::BrisaServiceXMLHandler::endElement(), and BrisaUpnp::BrisaServiceXMLHandler::startElement().

### 6.27.3 Member Data Documentation

**6.27.3.1 saxParserState BrisaUpnp::BrisaServiceParserContext::state**

Referenced by BrisaUpnp::BrisaServiceXMLHandler::characters(), BrisaUpnp::BrisaServiceXMLHandler::endElement(), BrisaUpnp::BrisaServiceXMLHandler::parseService(), and BrisaUpnp::BrisaServiceXMLHandler::startElement().

**6.27.3.2 int BrisaUpnp::BrisaServiceParserContext::stateSkip**

Referenced by BrisaUpnp::BrisaServiceXMLHandler::endElement(), BrisaUpnp::BrisaServiceXMLHandler::parseService(), and BrisaUpnp::BrisaServiceXMLHandler::startElement().

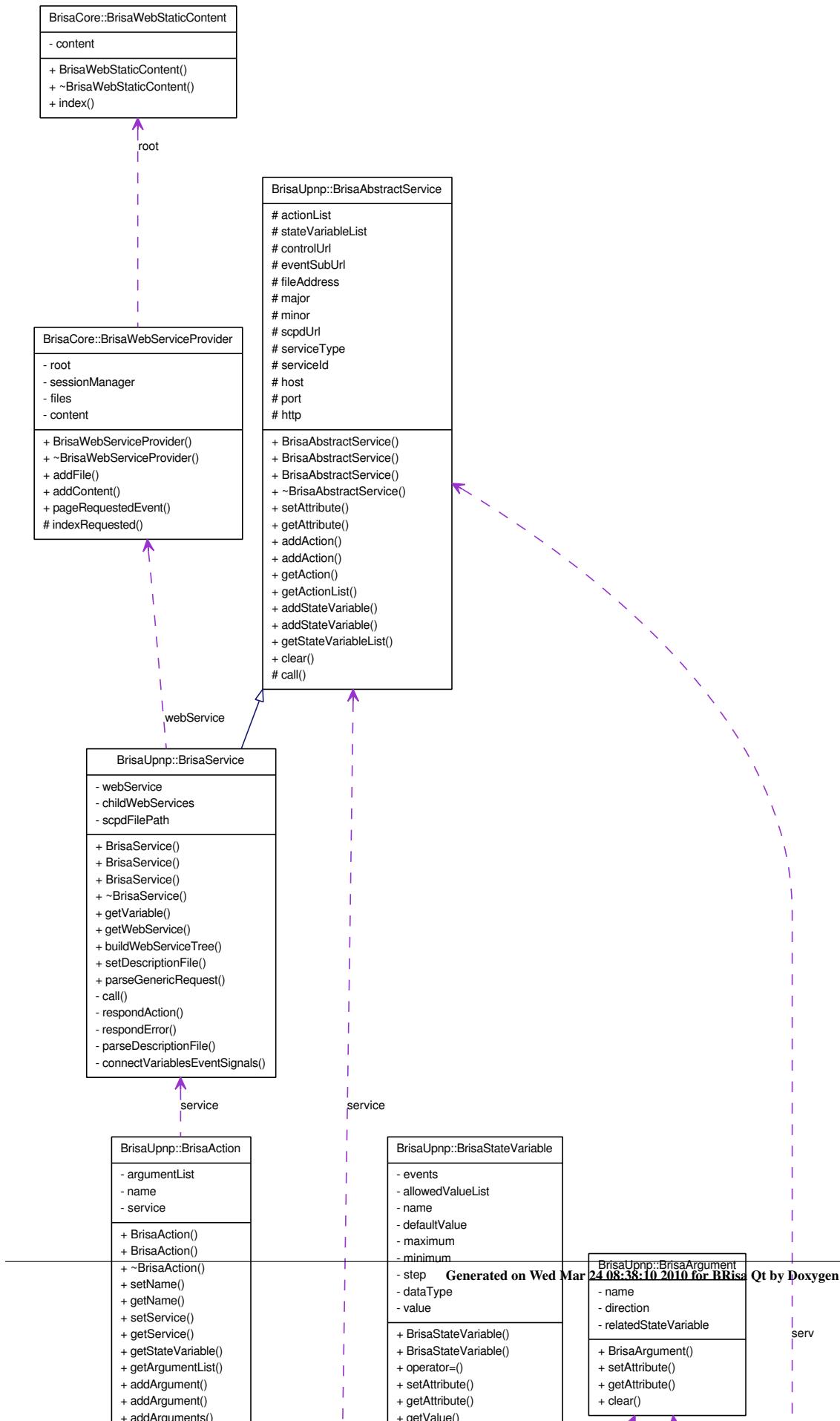
The documentation for this class was generated from the following file:

- [src/upnp/brisaservicexmlhandler.h](#)

## 6.28 BrisaUpnp::BrisaServiceXMLHandler Class Reference

#include <brisaservicexmlhandler.h>Collaboration diagram for

## BrisaUpnp::BrisaServiceXMLHandler:



## Public Member Functions

- void [parseService](#) ([BrisaAbstractService](#) \*service, [QIODevice](#) \*scpd)

## Protected Member Functions

- bool [startElement](#) (const [QString](#) &namespaceURI, const [QString](#) &localName, const [QString](#) &qName, const [QXmlAttributes](#) &attributes)
- bool [endElement](#) (const [QString](#) &namespaceURI, const [QString](#) &localName, const [QString](#) &qName)
- bool [characters](#) (const [QString](#) &str)

### 6.28.1 Member Function Documentation

**6.28.1.1 bool BrisaServiceXMLHandler::characters (const QString & str) [protected]**

**6.28.1.2 bool BrisaServiceXMLHandler::endElement (const QString & namespaceURI, const QString & localName, const QString & qName) [protected]**

**6.28.1.3 void BrisaServiceXMLHandler::parseService (BrisaAbstractService \* service, QIODevice \* scpd)**

Referenced by [BrisaUpnp::BrisaControlPointService::parseFromXml\(\)](#).

**6.28.1.4 bool BrisaServiceXMLHandler::startElement (const QString & namespaceURI, const QString & localName, const QString & qName, const QXmlAttributes & attributes) [protected]**

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

- [src/upnp/brisaservicexmlhandler.h](#)
- [src/upnp/brisaservicexmlhandler.cpp](#)

## 6.29 BrisaUpnp::BrisaSSDPClient Class Reference

SSDP stack implementantion for UPnP control points.

```
#include <BrisaUpnp/BrisaSSDPClient>
```

### Public Slots

- void [start \(\)](#)  
*Connects to the MultiCast group and starts the client.*
- void [stop \(\)](#)  
*Stops the client.*
- bool [isRunning \(\) const](#)  
*Returns true if the client is running.*

### Signals

- void [newDeviceEvent \(const QString &usn, const QString &location, const QString &st, const QString &ext, const QString &server, const QString &cacheControl\)](#)  
*This signal is emitted when the client receives a "ssdp:alive" message from a device joining the network.*
- void [removedDeviceEvent \(const QString &usn\)](#)  
*This signal is emitted when the client receives a "ssdp:byebye" message from a device leaving the network.*

### Public Member Functions

- [BrisaSSDPClient \(QObject \\*parent=0\)](#)  
*Constructs a BrisaSSCPClient with the given parent.*
- [~BrisaSSDPClient \(\)](#)  
*Destroys the client.*

#### 6.29.1 Detailed Description

SSDP stack implementantion for UPnP control points. Create a new BrisaSSCPClient and call "start()" to connect to the multicast group and start listening to ssdp notification messages.

When [BrisaSSDPClient](#) receives a notification message it emits "[newDeviceEvent\(\)](#)" in case of "ssdp:alive" and "[removedDeviceEvent](#)" in case of "ssdp:byebye". Other ssdp messages will be ignored.

#### 6.29.2 Constructor & Destructor Documentation

##### 6.29.2.1 BrisaSSDPClient::BrisaSSDPClient (QObject \* parent = 0)

Constructs a BrisaSSCPClient with the given parent.

### 6.29.2.2 BrisaSSDPClient::~BrisaSSDPClient ()

Destroys the client. Stops the client if it's running.

## 6.29.3 Member Function Documentation

### 6.29.3.1 bool BrisaSSDPClient::isRunning () const [slot]

Returns true if the client is running.

Referenced by start(), stop(), and ~BrisaSSDPClient().

### 6.29.3.2 void BrisaSSDPClient::newDeviceEvent (const QString & usn, const QString & location, const QString & st, const QString & ext, const QString & server, const QString & cacheControl) [signal]

This signal is emitted when the client receives a "ssdp:alive" message from a device joining the network.

See also:

[removedDeviceEvent\(\)](#)

### 6.29.3.3 void BrisaSSDPClient::removedDeviceEvent (const QString & usn) [signal]

This signal is emitted when the client receives a "ssdp:byebye" message from a device leaving the network.

See also:

[newDeviceEvent\(\)](#)

### 6.29.3.4 void BrisaSSDPClient::start () [slot]

Connects to the MultiCast group and starts the client.

See also:

[isRunning\(\)](#), [stop\(\)](#)

Referenced by BrisaUpnp::BrisaControlPoint::start().

### 6.29.3.5 void BrisaSSDPClient::stop () [slot]

Stops the client.

See also:

[isRunning\(\)](#), [start\(\)](#)

Referenced by BrisaUpnp::BrisaControlPoint::stop(), and ~BrisaSSDPClient().

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

- src/upnp/ssdp/[brisassdpclient.h](#)
- src/upnp/ssdp/[brisassdpclient.cpp](#)

## 6.30 BrisaUpnp::BrisaSSDPServer Class Reference

SSDP stack implementation for UPnP devices.

```
#include <BrisaUpnp/BrisaSSDPServer>
```

### Public Slots

- bool **isRunning** ()

*Returns true if [BrisaSSDPServer](#) is running.*

- void **start** ()

*Call this method to join the multicast group and start listening for UPnP msearch responses.*

- void **stop** ()

*Stops the [BrisaSSDPServer](#).*

- void **doNotify** (const QString &usn, const QString &location, const QString &st, const QString &server, const QString &cacheControl)

*Sends a UPnP notify alive message to the multicast group with the given information.*

- void **doByeBye** (const QString &usn, const QString &st)

*Sends a UPnP notify byebye message to the multicast group with the given information.*

- void **respondMSearch** (const QString &senderIp, quint16 senderPort, const QString &cacheControl, const QString &date, const QString &location, const QString &server, const QString &st, const QString &usn)

*Sends a UPnP msearch response message to the given sender IP address and port.*

### Signals

- void **msearchRequestReceived** (const QString &st, const QString &senderIp, quint16 senderPort)

*This signal is emitted when the [BrisaSSDPServer](#) receives a valid UPnP msearch request.*

### Public Member Functions

- **BrisaSSDPServer** (QObject \*parent=0)

*Constructs a [BrisaSSDPServer](#) with the given parent object.*

- virtual **~BrisaSSDPServer** ()

*Destroys the Object.*

### 6.30.1 Detailed Description

SSDP stack implementation for UPnP devices. Call [\*start\(\)\*](#) to begin listening for MSearch requests from control points. Whenever a new msearch request is parsed by the [BrisaSSDPServer](#), a [\*msearchRequestReceived\(\)\*](#) signal is emitted containing all of the request information. You can connect this signal to some slot which calls [\*respondMSearch\(\)\*](#) and get a synchronous response to msearch requests.

[BrisaSSDPServer](#) also implements SSDP notify messages. Call [\*doNotify\(\)\*](#) or [\*doByeBye\(\)\*](#) when entering or leaving the multicast group.

### 6.30.2 Constructor & Destructor Documentation

#### 6.30.2.1 BrisaSSDPServer::BrisaSSDPServer (QObject \**parent* = 0)

Constructs a [BrisaSSDPServer](#) with the given parent object.

#### 6.30.2.2 BrisaSSDPServer::~BrisaSSDPServer () [virtual]

Destroys the Object. Stops the server if running.

### 6.30.3 Member Function Documentation

#### 6.30.3.1 void BrisaSSDPServer::doByeBye (const QString &*usn*, const QString &*st*) [slot]

Sends a UPnP notify byebye message to the multicast group with the given information.

See also:

[doNotify\(\)](#)

Referenced by BrisaUpnp::BrisaDevice::doByeBye().

#### 6.30.3.2 void BrisaSSDPServer::doNotify (const QString &*usn*, const QString &*location*, const QString &*st*, const QString &*server*, const QString &*cacheControl*) [slot]

Sends a UPnP notify alive message to the multicast group with the given information.

See also:

[doByeBye\(\)](#)

Referenced by BrisaUpnp::BrisaDevice::doNotify().

#### 6.30.3.3 bool BrisaSSDPServer::isRunning () [slot]

Returns true if [BrisaSSDPServer](#) is running.

Referenced by [start\(\)](#), [stop\(\)](#), and [~BrisaSSDPServer\(\)](#).

**6.30.3.4 void BrisaSSDPServer::msearchRequestReceived (const QString & *st*, const QString & *senderIp*, quint16 *senderPort*) [signal]**

This signal is emitted when the [BrisaSSDPServer](#) receives a valid UPnP msearch request.

**See also:**

[respondMSearch\(\)](#)

**6.30.3.5 void BrisaSSDPServer::respondMSearch (const QString & *senderIp*, quint16 *senderPort*, const QString & *cacheControl*, const QString & *date*, const QString & *location*, const QString & *server*, const QString & *st*, const QString & *usn*) [slot]**

Sends a UPnP msearch response message to the given sender IP address and port. Connect this slot to a proper signal to get synchronous response for msearch requests.

**See also:**

[msearchRequestReceived\(\)](#)

Referenced by [BrisaUpnp::BrisaDevice::respondMSearch\(\)](#).

**6.30.3.6 void BrisaSSDPServer::start () [slot]**

Call this method to join the multicast group and start listening for UPnP msearch responses.

**See also:**

[stop\(\)](#)

Referenced by [BrisaUpnp::BrisaDevice::start\(\)](#).

**6.30.3.7 void BrisaSSDPServer::stop () [slot]**

Stops the [BrisaSSDPServer](#).

**See also:**

[start\(\)](#)

Referenced by [BrisaUpnp::BrisaDevice::stop\(\)](#), and [~BrisaSSDPServer\(\)](#).

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

- [src/upnp/ssdp/brisassdpserver.h](#)
- [src/upnp/ssdp/brisassdpserver.cpp](#)

## 6.31 BrisaUpnp::BrisaStateVariable Class Reference

Represents the service's state variables.

```
#include <BrisaUpnp/BrisaStateVariable>
```

### Public Types

- enum `BrisaStateVariableAttribute` {  
    Name, SendEvents, DataType, DefaultValue,  
    AllowedValue, Minimum, Maximum, Step,  
    Value }

### Signals

- void `changed (BrisaStateVariable *)`

### Public Member Functions

- `BrisaStateVariable (QString sendEvents="", QString name="", QString datatype="", QString defaultValue="", QString maximum="", QString minimum="", QString step="", QObject *parent=0)`  
*Constructs a `BrisaStateVariable` that sendEvents with the given name, datatype, defaultValue, maximum value, minimum value, value step, and parent.*
- `BrisaStateVariable (const BrisaStateVariable &)`  
*Constructs a `BrisaStateVariable` from the given variable.*
- `BrisaStateVariable & operator= (const BrisaStateVariable &)`  
*Sets this variable equals to variable.*
- void `setAttribute (BrisaStateVariableAttribute attr, QVariant value)`  
*Sets its attribute attr to the given value.*
- `QString getAttribute (BrisaStateVariableAttribute attr, int index=0) const`  
*Returns attr value as a `QString`.*
- `QVariant getValue () const`  
*Returns the stored value as a `QVariant`.*
- `QVariant::Type getDataType () const`
- bool `sendEvents () const`  
*Returns true if the variable is set to send events.*
- void `addAllowedValue (QString allowedValue)`  
*Adds a value to the list of values that can be set to its Value attribute.*
- `QList< QString > getAllowedValueList ()`  
*Returns the list of values that can be set to its Value attribute.*

- void [clear \(\)](#)

*Clears this variable's attributtes.*

### 6.31.1 Detailed Description

Represents the service's state variables.

### 6.31.2 Member Enumeration Documentation

#### 6.31.2.1 enum BrisaUpnp::BrisaStateVariable::BrisaStateVariableAttribute

Enumerator:

*Name*  
*SendEvents*  
*DataType*  
*DefaultValue*  
*AllowedValue*  
*Minimum*  
*Maximum*  
*Step*  
*Value*

### 6.31.3 Constructor & Destructor Documentation

#### 6.31.3.1 BrisaStateVariable::BrisaStateVariable (QString *sendEvents* = "", QString *name* = "", QString *datatype* = "", QString *defaultValue* = "", QString *maximum* = "", QString *minimum* = "", QString *step* = "", QObject \**parent* = 0)

Constructs a [BrisaStateVariable](#) that *sendEvents* with the given *name*, *datatype*, *defaultValue*, *maximum* value, *minimum* value, value *step*, and *parent*.

#### 6.31.3.2 BrisaStateVariable::BrisaStateVariable (const BrisaStateVariable & *variable*)

Constructs a [BrisaStateVariable](#) from the given *variable*.

### 6.31.4 Member Function Documentation

#### 6.31.4.1 void BrisaStateVariable::addAllowedValue (QString *allowedValue*)

Adds a value to the list of values that can be set to its Value attribute.

Referenced by BrisaUpnp::BrisaServiceXMLHandler::characters().

#### 6.31.4.2 void BrisaUpnp::BrisaStateVariable::changed (BrisaStateVariable \*) [signal]

Referenced by setAttribute().

**6.31.4.3 void BrisaStateVariable::clear ()**

Clears this variable's attributtes.

**6.31.4.4 QList< QString > BrisaStateVariable::getAllowedValueList ()**

Returns the list of values that can be set to its Value attribute.

**6.31.4.5 QString BrisaStateVariable::getAttribute (BrisaStateVariableAttribute *attr*, int *index* = 0) const**

Returns *attr* value as a QString.

Referenced by BrisaStateVariable(), and operator=( ).

**6.31.4.6 QVariant::Type BrisaStateVariable::getDataType () const****6.31.4.7 QVariant BrisaStateVariable::getValue () const**

Returns the stored value as a QVariant.

Referenced by BrisaStateVariable(), and operator=( ).

**6.31.4.8 BrisaStateVariable & BrisaStateVariable::operator= (const BrisaStateVariable & *variable*)**

Sets this variable equals to *variable*.

**6.31.4.9 bool BrisaStateVariable::sendEvents () const**

Returns true if the variable is set to send events.

Referenced by BrisaStateVariable(), and operator=( ).

**6.31.4.10 void BrisaStateVariable::setAttribute (BrisaStateVariableAttribute *attr*, QVariant *value*)**

Sets its attribute *attr* to the given *value*.

Referenced by BrisaUpnp::BrisaServiceXMLHandler::characters(), and BrisaUpnp::BrisaServiceXMLHandler::startElement().

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

- src/upnp/[brisastatevariable.h](#)
- src/upnp/[brisastatevariable.cpp](#)



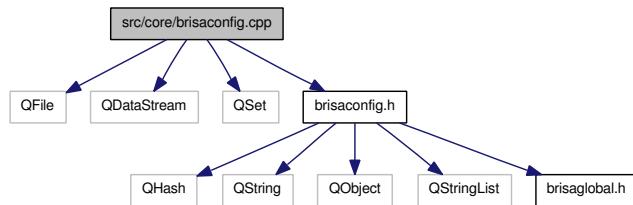
# Chapter 7

## File Documentation

### 7.1 src/core/brisacconfig.cpp File Reference

```
#include <QFile>
#include <QDataStream>
#include <QSet>
#include "brisacconfig.h"
```

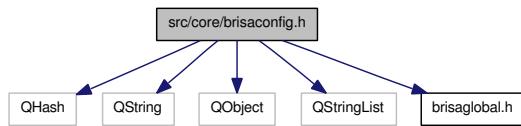
Include dependency graph for brisacconfig.cpp:



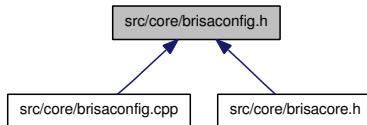
## 7.2 src/core/brisacconfig.h File Reference

```
#include <QHash>
#include <QString>
#include <QObject>
#include <QStringList>
#include "brisaglobal.h"
```

Include dependency graph for brisacconfig.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaCore::BrisaConfigurationManager](#)  
*Class that provides an easy way of managing configurations.*

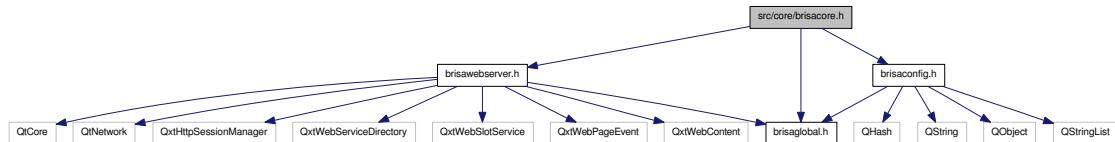
## Namespaces

- namespace [BrisaCore](#)

## 7.3 src/core/brisacore.h File Reference

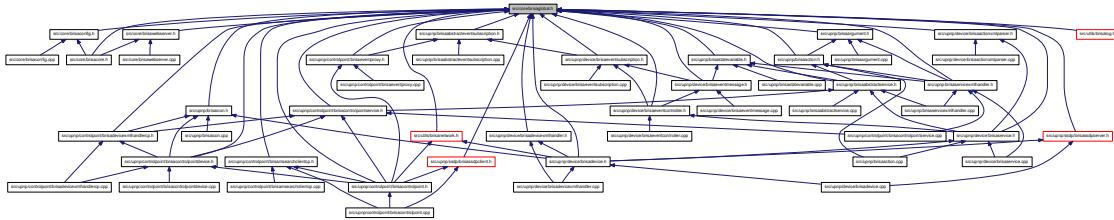
```
#include "brisawebserver.h"
#include "brisacconfig.h"
#include "brisaglobal.h"
```

Include dependency graph for brisacore.h:



## 7.4 src/core/brisaglobal.h File Reference

This graph shows which files directly or indirectly include this file:



### Defines

- #define **BRISA\_VERSION** 0x000001
- #define **BRISA\_VERSION\_STR** "0.1"
- #define **BRISA\_CORE\_EXPORT** Q\_DECL\_IMPORT
- #define **BRISA\_UTILS\_EXPORT** Q\_DECL\_IMPORT
- #define **BRISA\_UPNP\_EXPORT** Q\_DECL\_IMPORT

#### 7.4.1 Define Documentation

7.4.1.1 #define **BRISA\_CORE\_EXPORT** Q\_DECL\_IMPORT

7.4.1.2 #define **BRISA\_UPNP\_EXPORT** Q\_DECL\_IMPORT

7.4.1.3 #define **BRISA\_UTILS\_EXPORT** Q\_DECL\_IMPORT

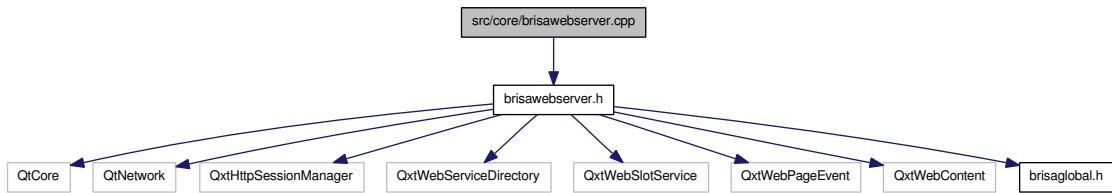
7.4.1.4 #define **BRISA\_VERSION** 0x000001

7.4.1.5 #define **BRISA\_VERSION\_STR** "0.1"

## 7.5 src/core/brisawebserver.cpp File Reference

```
#include "brisawebserver.h"
```

Include dependency graph for brisawebserver.cpp:



## Functions

- static QString [extractPathLevel](#) (QxtWebRequestEvent \*event)

### 7.5.1 Function Documentation

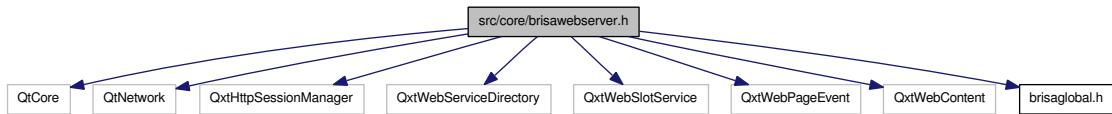
#### 7.5.1.1 static QString extractPathLevel (QxtWebRequestEvent \* *event*) [static]

Referenced by BrisaCore::BrisaServiceProvider::pageRequestedEvent().

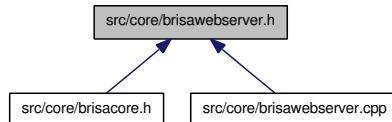
## 7.6 src/core/brisawebserver.h File Reference

```
#include <QtCore>
#include <QtNetwork>
#include <QxtHttpSessionManager>
#include <QxtWebServiceDirectory>
#include <QxtWebSlotService>
#include <QxtWebPageEvent>
#include <QxtWebContent>
#include "brisaglobal.h"
```

Include dependency graph for brisawebserver.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaCore::BrisaWebService](#)  
*Web service abstraction class.*
- class [BrisaCore::BrisaWebFile](#)  
*Adds a file to the web server.*
- class [BrisaCore::BrisaWebStaticContent](#)  
*The BrisaWebStaticContent class stores a QString into the web server.*
- class [BrisaCore::BrisaWebServiceProvider](#)  
*The BrisaWebServiceProvider class works as web service manager for the web server.*
- class [BrisaCore::BrisaWebserver](#)  
*The BrisaWebserver class is a web server implementation.*

## Namespaces

- namespace [BrisaCore](#)

## Defines

- #define **DEFAULT\_PAGE** "<html><body><h1>BRisa WebServer!\n</body></html>"

### 7.6.1 Define Documentation

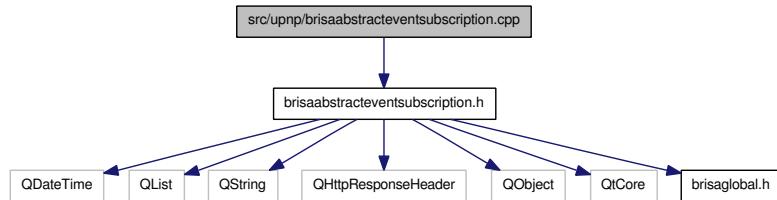
#### 7.6.1.1 #define **DEFAULT\_PAGE** "<html><body><h1>BRisa WebServer!\n</body></html>"

Referenced by BrisaCore::BrisaWebServiceProvider::BrisaWebServiceProvider(), and BrisaCore::BrisaWebService::pageRequestedEvent().

## 7.7 src/upnp/brisabSTRACTeventsUBSCRIPTION.cpp File Reference

```
#include "brisabSTRACTeventsUBSCRIPTION.h"
```

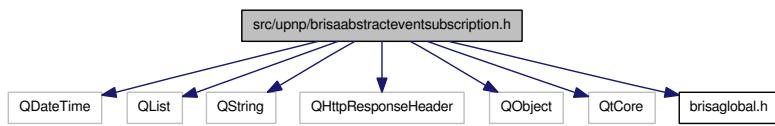
Include dependency graph for brisaabSTRACTeventsUBSCRIPTION.cpp:



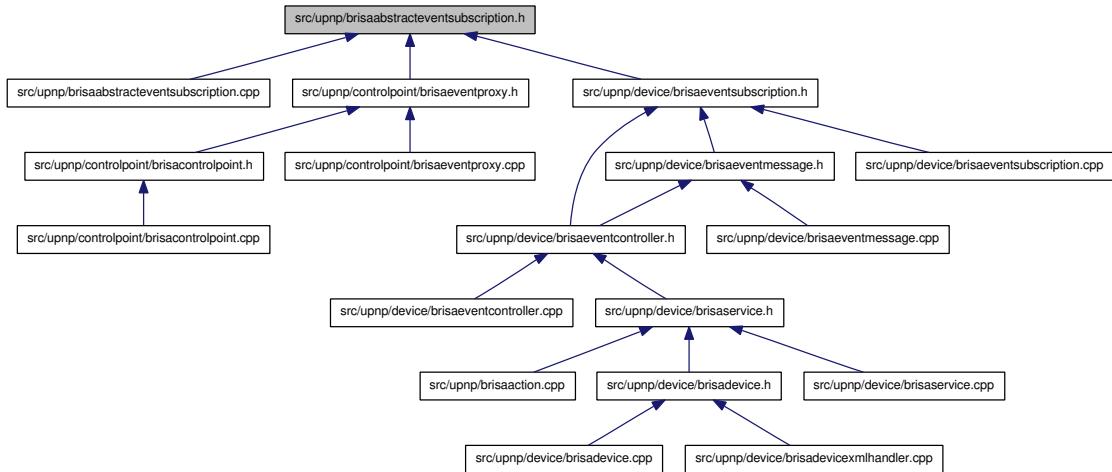
## 7.8 src/upnp/brisabSTRACTeventsSubscription.h File Reference

```
#include <QDateTime>
#include <QList>
#include <QString>
#include <QHttpResponseHeader>
#include <QObject>
#include <QtCore>
#include "brisaglobal.h"
```

Include dependency graph for brisaabSTRACTeventsSubscription.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaAbstractEventSubscription](#)

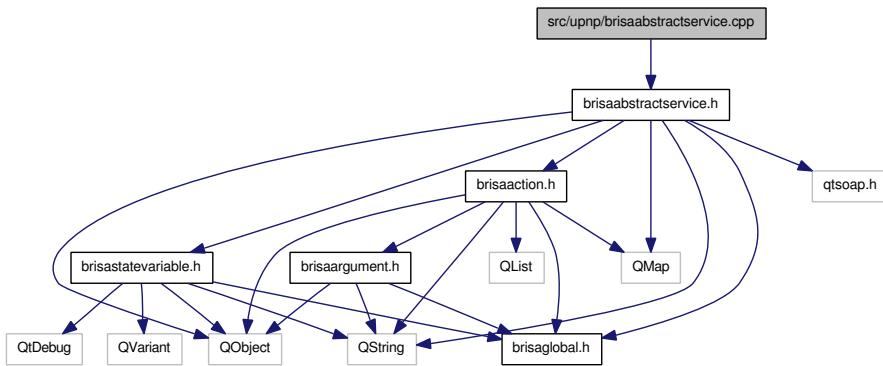
## Namespaces

- namespace [BrisaUpnp](#)

## 7.9 src/upnp/brisabSTRACTservice.cpp File Reference

```
#include "brisabSTRACTservice.h"
```

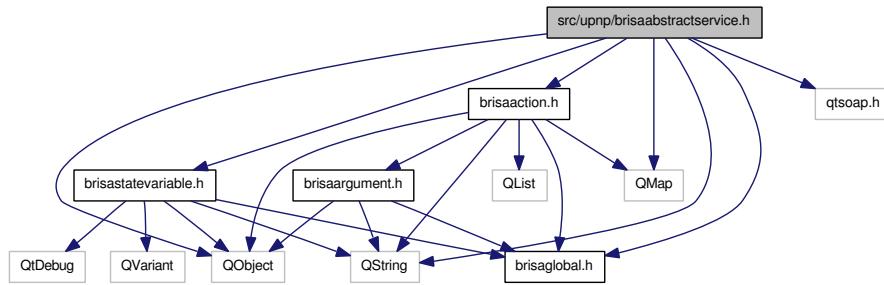
Include dependency graph for brisaabSTRACTservice.cpp:



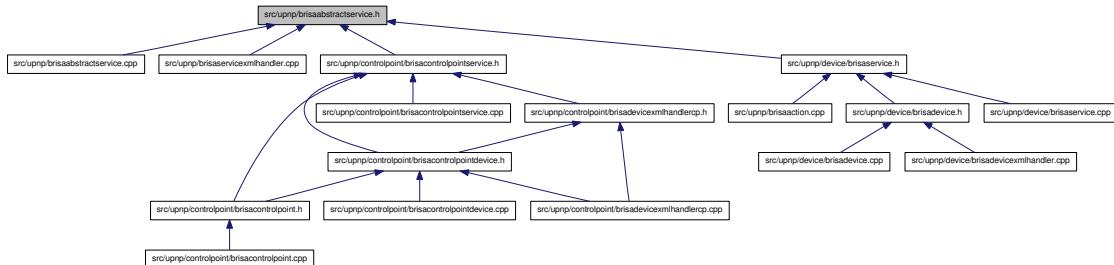
## 7.10 src/upnp/brisabSTRACTservice.h File Reference

```
#include "brisaction.h"
#include "brisastatevariable.h"
#include "brisaglobal.h"
#include <qtsoap.h>
#include <QMap>
#include <QString>
#include <QObject>
```

Include dependency graph for brisaabstractservice.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaAbstractService](#)

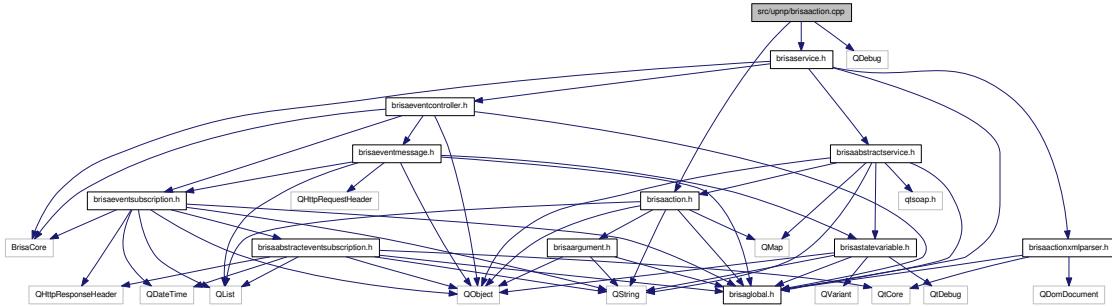
## Namespaces

- namespace [BrisaUpnp](#)

## 7.11 src/upnp/brisaction.cpp File Reference

```
#include "brisaction.h"
#include "brisaservice.h"
#include <QDebug>
```

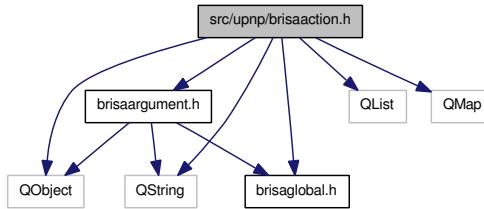
Include dependency graph for brisaaction.cpp:



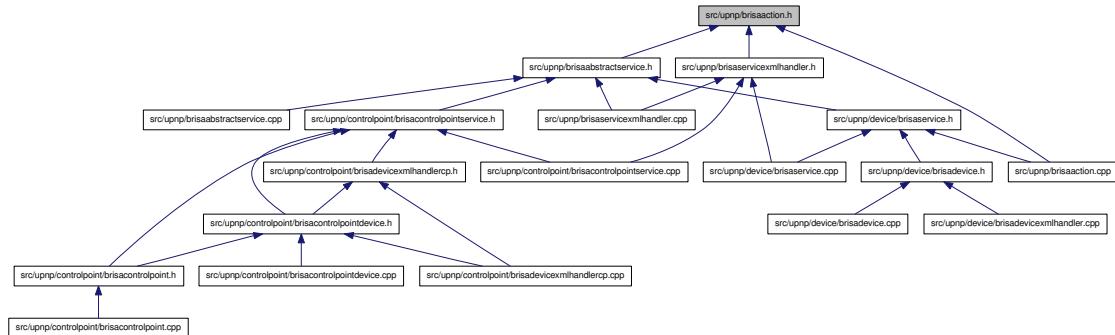
## 7.12 src/upnp/brisaction.h File Reference

```
#include "brisargument.h"
#include "brisaglobal.h"
#include <QString>
#include <QList>
#include <QMap>
#include <QObject>
```

Include dependency graph for brisaaction.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaAction](#)  
*Template method class that represents each service's action.*

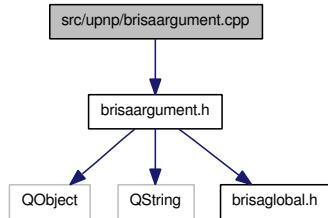
## Namespaces

- namespace [BrisaUpnp](#)

## 7.13 src/upnp/brisargument.cpp File Reference

```
#include "brisargument.h"
```

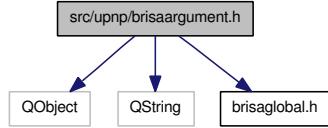
Include dependency graph for brisaargument.cpp:



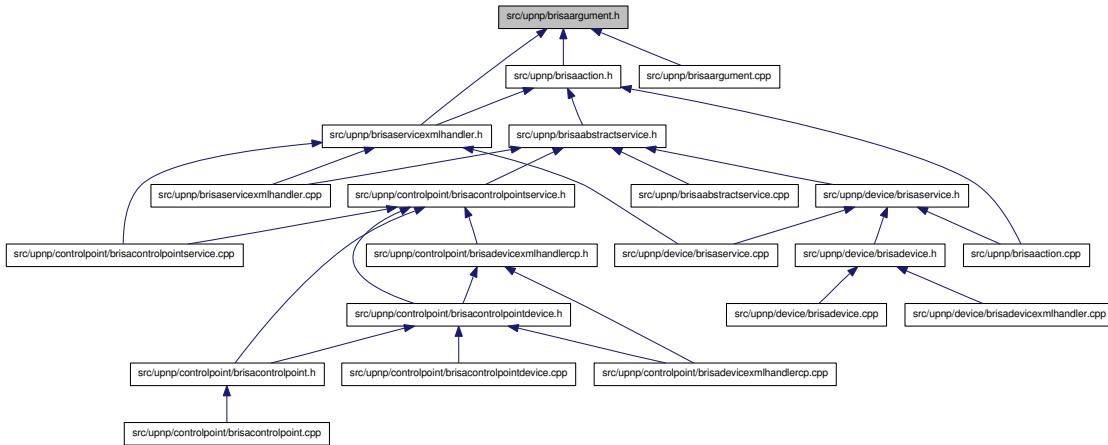
## 7.14 src/upnp/brisargument.h File Reference

```
#include <QObject>
#include <QString>
#include "brisaglobal.h"
```

Include dependency graph for brisaargument.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaArgument](#)

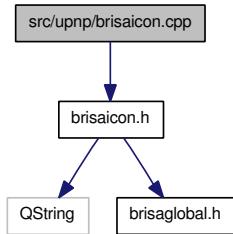
## Namespaces

- namespace [BrisaUpnp](#)

## 7.15 src/upnp/brisaticon.cpp File Reference

```
#include "brisaticon.h"
```

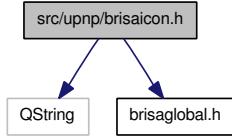
Include dependency graph for brisaicon.cpp:



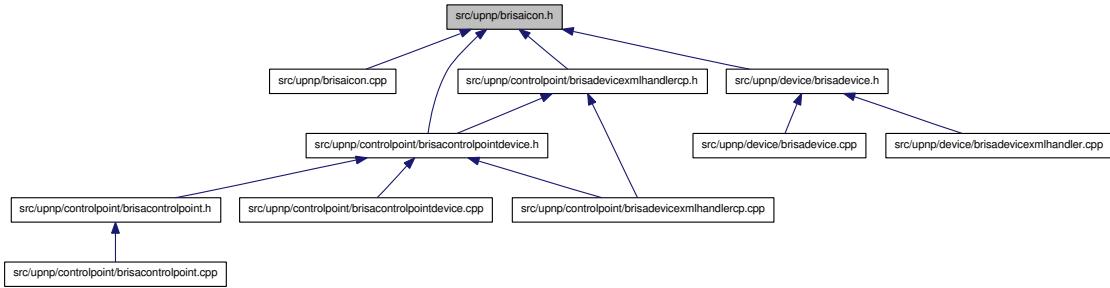
## 7.16 src/upnp/brisaticon.h File Reference

```
#include <QString>
#include "brisaglobal.h"
```

Include dependency graph for brisaticon.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaIcon](#)

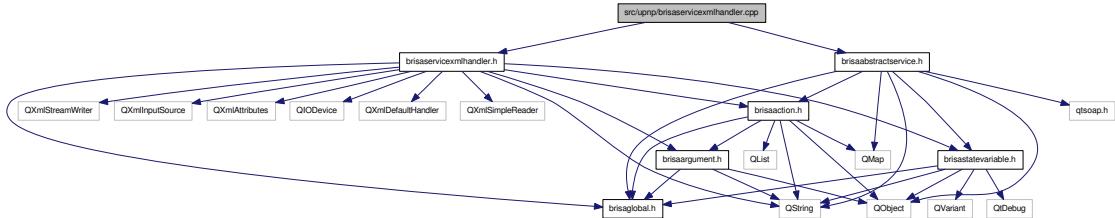
## Namespaces

- namespace [BrisaUpnp](#)

## 7.17 src/upnp/brisaservicexmlhandler.cpp File Reference

```
#include "brisaservicexmlhandler.h"
#include "brisabSTRACTservice.h"
```

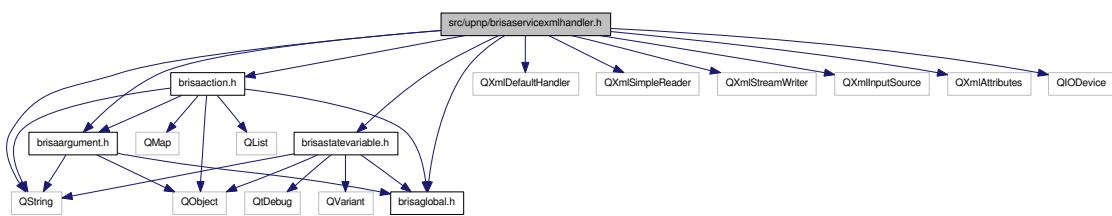
Include dependency graph for brisaservicexmlhandler.cpp:



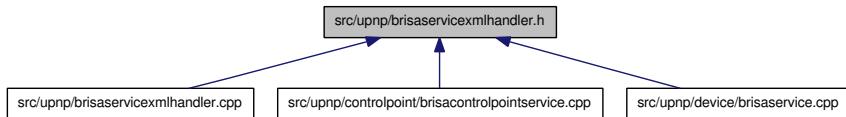
## 7.18 src/upnp/brisaservicexmlhandler.h File Reference

```
#include "brisaaction.h"
#include "brisargument.h"
#include "brisastatevariable.h"
#include "brisaglobal.h"
#include <QXmlDefaultHandler>
#include <QXmlSimpleReader>
#include <QXmlStreamWriter>
#include <QXmlInputSource>
#include <QXmlAttributes>
#include <QIODevice>
#include <QString>
```

Include dependency graph for brisaservicexmlhandler.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaServiceParserContext](#)
- class [BrisaUpnp::BrisaServiceXMLHandler](#)

## Namespaces

- namespace [BrisaUpnp](#)

## Enumerations

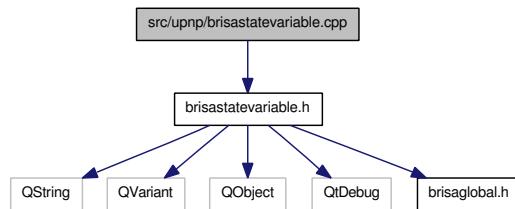
- enum [BrisaUpnp::saxParserState](#) {
   
    [BrisaUpnp::ServiceStart](#),                   [BrisaUpnp::Scpd](#),                   [BrisaUpnp::ServiceSpecVersion](#),
   
    [BrisaUpnp::ServiceSpecVersionMajor](#),
 }

```
BrisaUpnp::ServiceSpecVersionMinor,      BrisaUpnp::ActionList,      BrisaUpnp::Action,
BrisaUpnp::ActionName,
BrisaUpnp::ArgumentList,      BrisaUpnp::Argument,      BrisaUpnp::ArgumentName,
BrisaUpnp::ArgumentDirection,
BrisaUpnp::RelatedStateVariable,      BrisaUpnp::ServiceStateTable,      BrisaUpnp::StateVariable,
BrisaUpnp::StateVariableName,
BrisaUpnp::StateVariableDataType,      BrisaUpnp::StateVariableDefaultValue,
BrisaUpnp::StateVariableAllowedValueList, BrisaUpnp::StateVariableAllowedValue,
BrisaUpnp::StateVariableAllowedValueRange, BrisaUpnp::StateVariableAllowedValueRangeMinimum,
BrisaUpnp::StateVariableAllowedValueRangeMaximum, BrisaUpnp::StateVariableAllowedValueRangeStep,
BrisaUpnp::ServiceFinished, BrisaUpnp::ServiceError = -1 }
```

## 7.19 src/upnp/brisastatevariable.cpp File Reference

```
#include "brisastatevariable.h"
```

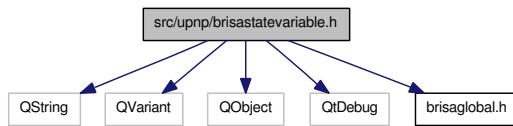
Include dependency graph for brisastatevariable.cpp:



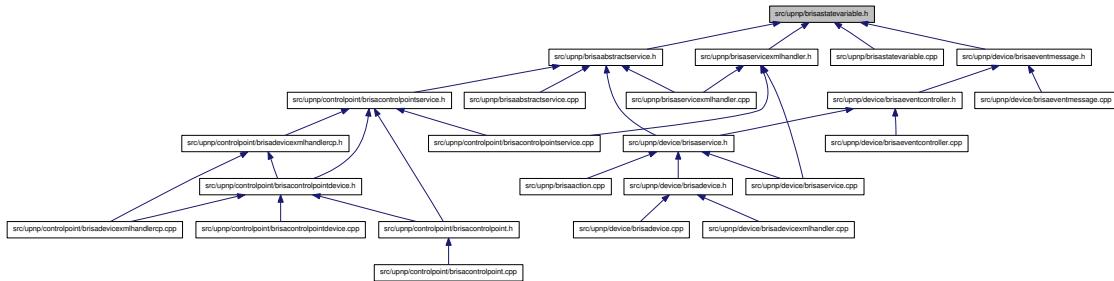
## 7.20 src/upnp/brisastatevariable.h File Reference

```
#include <QString>
#include <QVariant>
#include <QObject>
#include <QtDebug>
#include "brisaglobal.h"
```

Include dependency graph for brisastatevariable.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaStateVariable](#)  
*Represents the service's state variables.*

## Namespaces

- namespace [BrisaUpnp](#)

## 7.21 src/upnp/controlpoint/brisaccontrolpoint.cpp File Reference

```
#include <QtCore>
#include <QtDebug>
#include "brisaccontrolpoint.h"
#include "brisassdpclient.h"
#include "brisamsearchclientcp.h"
```

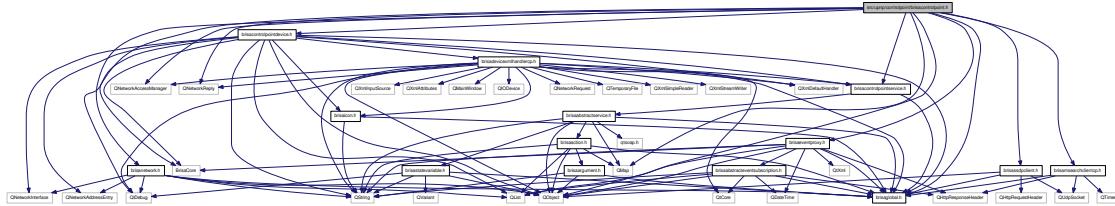
Include dependency graph for brisacontrolpoint.cpp:



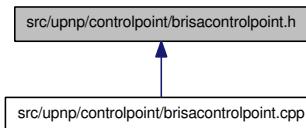
## 7.22 src/upnp/controlpoint/brisaccontrolpoint.h File Reference

```
#include <QNetworkAccessManager>
#include <QNetworkReply>
#include <QObject>
#include <QMap>
#include <BrisaCore>
#include "brisanetwork.h"
#include "brisaccontrolpointdevice.h"
#include "brisaccontrolpointservice.h"
#include "brisaeventproxy.h"
#include "brisamsearchclientcp.h"
#include "brisassdpclient.h"
#include "brisaglobal.h"
```

Include dependency graph for brisacontrolpoint.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaControlPoint](#)  
*Class that implements the control part in UPnP Architecture.*

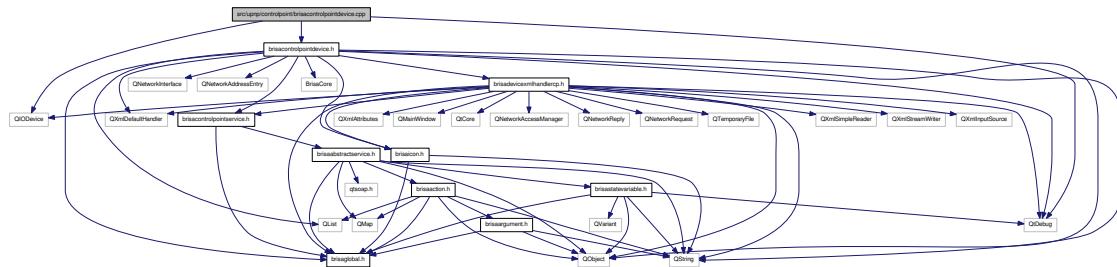
## Namespaces

- namespace [BrisaUpnp](#)

## **7.23 src/upnp/controlpoint/brisaccontrolpointdevice.cpp File Reference**

```
#include <QtDebug>
#include <QIODevice>
#include "brisaccontrolpointdevice.h"
```

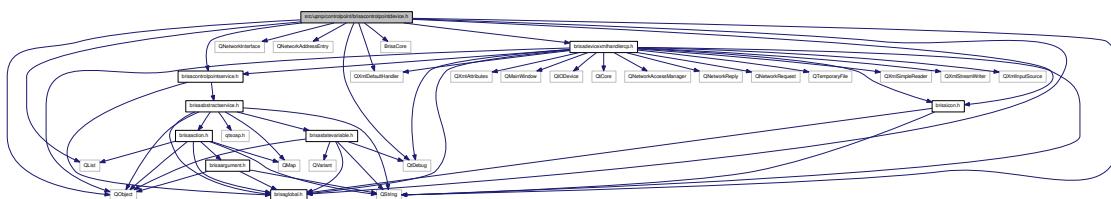
Include dependency graph for brisacontrolpointdevice.cpp:



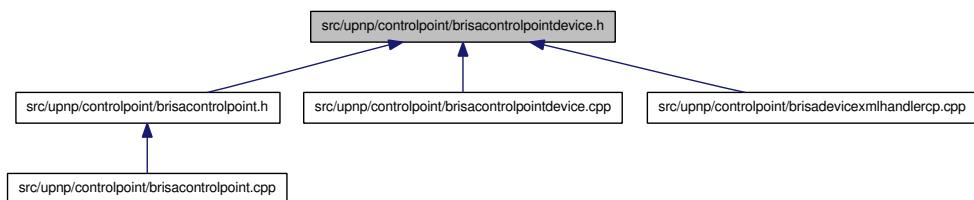
## 7.24 src/upnp/controlpoint/brisaccontrolpointdevice.h File Reference

```
#include <QString>
#include <QList>
#include <QXmlDefaultHandler>
#include <QNetworkInterface>
#include <QNetworkAddressEntry>
#include <QtDebug>
#include <QObject>
#include <BrisaCore>
#include "brisaiicon.h"
#include "brisicontrolpointservice.h"
#include "brisadevicexmlhandlercp.h"
#include "brisaglobal.h"
```

Include dependency graph for brisacontrolpointdevice.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class **BrisaUpnp::BrisaControlPointDevice**  
*Class that implements the devices that control point part is going to handle.*

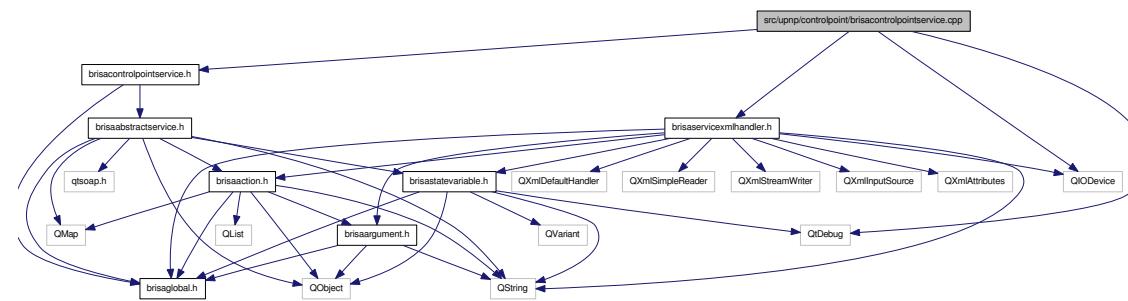
## Namespaces

- namespace BrisaUpnp

## 7.25 src/upnp/controlpoint/brisaccontrolpointservice.cpp File Reference

```
#include "brisaccontrolpointservice.h"
#include "brisaservicexmlhandler.h"
#include <QtDebug>
#include <QIODevice>
```

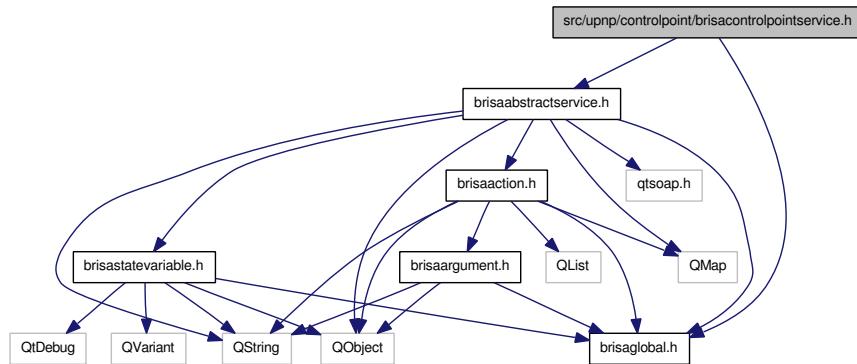
Include dependency graph for brisaccontrolpointservice.cpp:



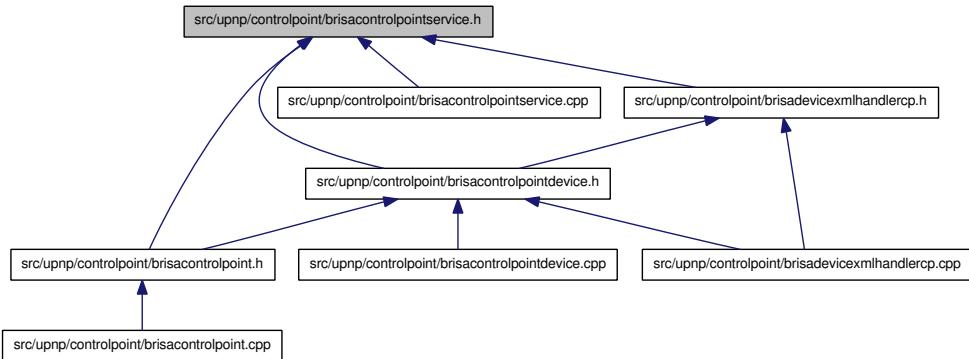
## 7.26 src/upnp/controlpoint/brisaccontrolpointservice.h File Reference

```
#include "brisababstractservice.h"
#include "brisaglobal.h"
```

Include dependency graph for brisacontrolpointservice.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaControlPointService](#)

*BrisaControlPointService* is the class that implements action control in UPnP Architecture.

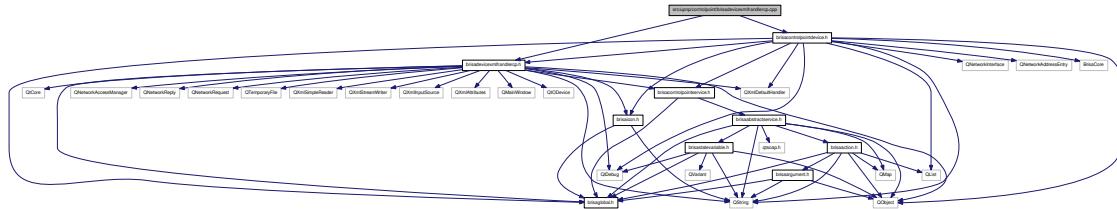
## Namespaces

- namespace [BrisaUpnp](#)

## **7.27 src/upnp/controlpoint/brisadevicexmlhandlercp.cpp File Reference**

```
#include "brisadevicexmlhandlercp.h"  
#include "brisicontrolpointdevice.h"
```

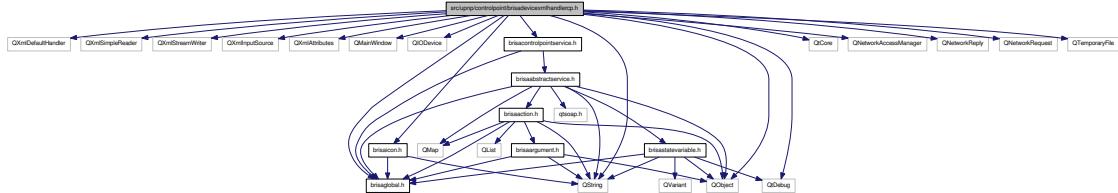
Include dependency graph for brisadevicexmlhandlercp.cpp:



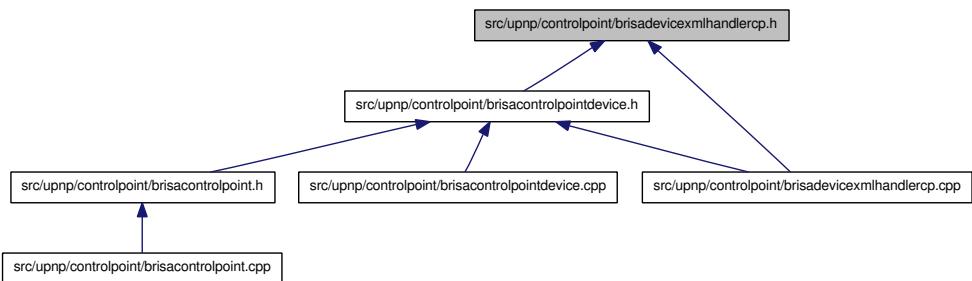
## **7.28 src/upnp/controlpoint/brisadevicexmlhandlercp.h File Reference**

```
#include <QXmlDefaultHandler>
#include <QXmlSimpleReader>
#include <QXmlStreamWriter>
#include <QXmlInputSource>
#include <QXmlAttributes>
#include <QMainWindow>
#include <QIODevice>
#include <QString>
#include <QtDebug>
#include <QtCore>
#include <QObject>
#include <QNetworkAccessManager>
#include <QNetworkReply>
#include <QNetworkRequest>
#include <QTemporaryFile>
#include "brisaticon.h"
#include "brisicontrolpointservice.h"
#include "brisaglobal.h"
```

Include dependency graph for brisadevicexmlhandlercp.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaDeviceParserContext](#)
- class [BrisaUpnp::BrisaDeviceXMLHandlerCP](#)  
*BrisaDeviceXMLHandlerCP creates a device from a xml description file, with all it's attributes, it let's it ready to be used.*
- class [BrisaUpnp::BrisaServiceFetcher](#)

## Namespaces

- namespace [BrisaUpnp](#)

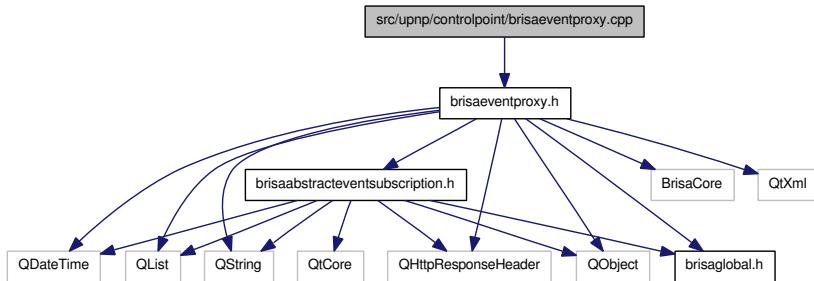
## Enumerations

- enum [BrisaUpnp::SaxParserState](#) {  
    BrisaUpnp::Start, BrisaUpnp::Root, BrisaUpnp::SpecVersion, BrisaUpnp::SpecVersionMajor,  
    BrisaUpnp::SpecVersionMinor, BrisaUpnp::UrlBase, BrisaUpnp::Device, BrisaUpnp::DeviceType,  
    BrisaUpnp::DeviceFriendlyName,     BrisaUpnp::Manufacturer,     BrisaUpnp::ManufacturerUrl,  
    BrisaUpnp::ModelDescription,  
    BrisaUpnp::modelName, BrisaUpnp::ModelUrl, BrisaUpnp::SerialNumber, BrisaUpnp::Udn,  
    BrisaUpnp::Upc, BrisaUpnp::IconList, BrisaUpnp::Icon, BrisaUpnp::IconMimetype,  
    BrisaUpnp::IconWidth, BrisaUpnp::IconHeight, BrisaUpnp::IconDepth, BrisaUpnp::IconUrl,  
    BrisaUpnp::PresentationUrl, BrisaUpnp::DeviceList, BrisaUpnp::ServiceList, BrisaUpnp::Service,  
    BrisaUpnp::ServiceType,                 BrisaUpnp::ServiceId,                 BrisaUpnp::ServiceScpdUrl,  
    BrisaUpnp::ServiceControlUrl,  
    BrisaUpnp::ServiceEventSubUrl, BrisaUpnp::Finished, BrisaUpnp::Error = -1 }

## 7.29 src/upnp/controlpoint/brisaeventproxy.cpp File Reference

```
#include "brisaeventproxy.h"
```

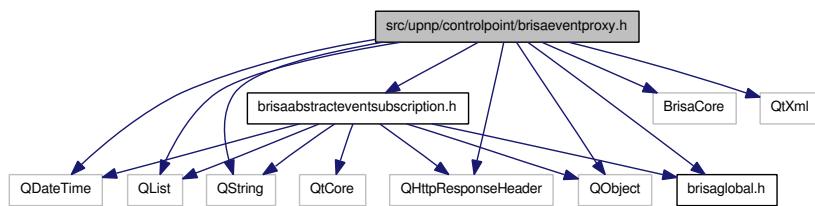
Include dependency graph for brisaeventproxy.cpp:



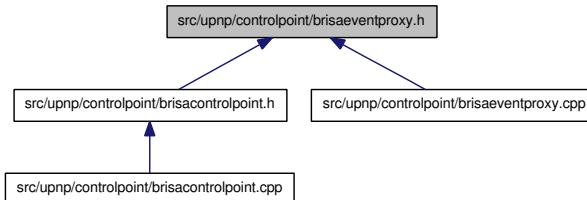
## 7.30 src/upnp/controlpoint/brisaeventproxy.h File Reference

```
#include "brisabSTRACTeventsSubscription.h"
#include "brisaglobal.h"
#include <BrisaCore>
#include <QDateTime>
#include <QList>
#include <QString>
#include <QHttpResponseHeader>
#include <QObject>
#include <QtXml>
```

Include dependency graph for brisaeventproxy.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaEventProxy](#)

*Class that implements the event part in control point side in Brisa Qt, this class makes the operations of subscribe, renew subscription and unsubscribe.*

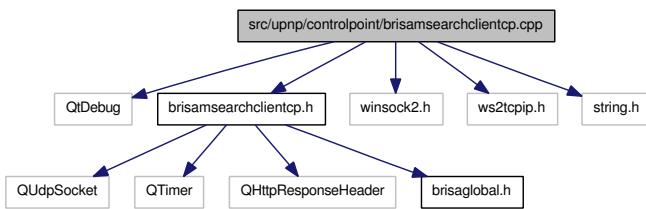
## Namespaces

- namespace [BrisaUpnp](#)

## 7.31 src/upnp/controlpoint/brisamsearchclientcp.cpp File Reference

```
#include <QtDebug>
#include "brisamsearchclientcp.h"
#include <winsock2.h>
#include <ws2tcpip.h>
#include <string.h>
```

Include dependency graph for brisamsearchclientcp.cpp:



### Defines

- #define UPNP\_MSEARCH\_DISCOVER

#### 7.31.1 Define Documentation

##### 7.31.1.1 #define UPNP\_MSEARCH\_DISCOVER

**Value:**

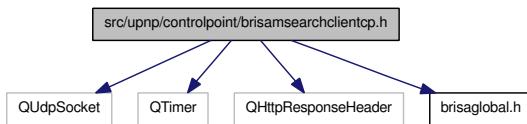
```
"M-SEARCH * HTTP/1.1\r\n"
                                \
"HOST: 239.255.255.250:1900\r\n" \
"MAN: \"ssdp:discover\"\r\n"      \
"MX: %1\r\n"                      \
"ST: %2\r\n"                      \
"\r\n"
```

Referenced by BrisaUpnp::BrisaMSearchClientCP::discover().

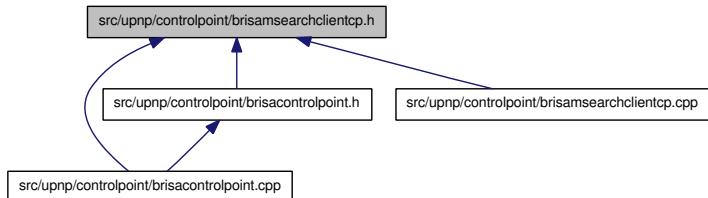
## 7.32 src/upnp/controlpoint/brisamsearchclientcp.h File Reference

```
#include <QUdpSocket>
#include <QTimer>
#include <QHttpResponseHeader>
#include "brisaglobal.h"
```

Include dependency graph for brisamsearchclientcp.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class [BrisaUpnp::BrisaMSearchClientCP](#)  
*SSDP MSearch implementation for UPnP control points.*

### Namespaces

- namespace [BrisaUpnp](#)

### Defines

- #define [DEFAULT\\_SEARCH\\_TIME](#) 600
- #define [DEFAULT\\_SEARCH\\_TYPE](#) "ssdp:all"

#### 7.32.1 Define Documentation

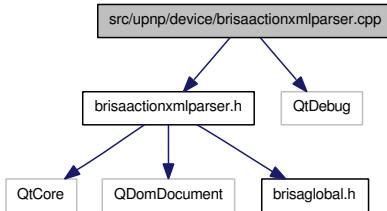
##### 7.32.1.1 #define DEFAULT\_SEARCH\_TIME 600

##### 7.32.1.2 #define DEFAULT\_SEARCH\_TYPE "ssdp:all"

### 7.33 src/upnp/device/brisactionxmlparser.cpp File Reference

```
#include "brisactionxmlparser.h"
#include <QtDebug>
```

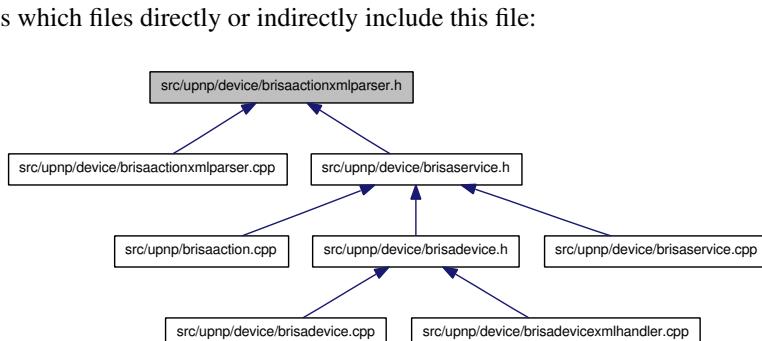
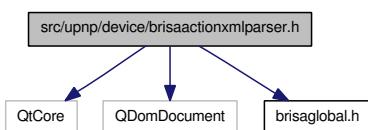
Include dependency graph for brisactionxmlparser.cpp:



## 7.34 src/upnp/device/brisactionxmlparser.h File Reference

```
#include <QtCore>
#include <QDomDocument>
#include "brisaglobal.h"
```

Include dependency graph for brisaactionxmlparser.h:



## Classes

- class [BrisaUpnp::BrisaActionXmlParser](#)

*XML parser for SOAP requests.*

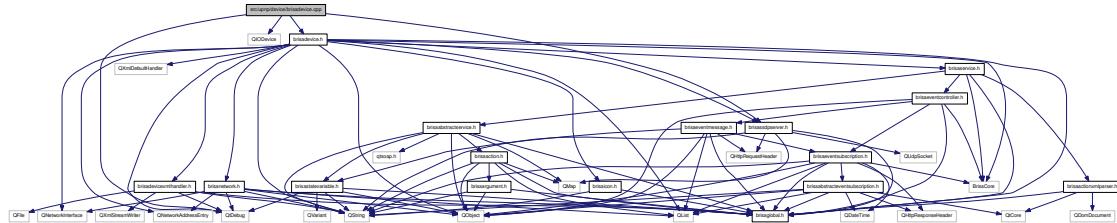
## Namespaces

- namespace [BrisaUpnp](#)

## 7.35 src/upnp/device/brisadevice.cpp File Reference

```
#include <QtDebug>
#include <QIODevice>
#include "brisadevice.h"
#include "brisassdpserver.h"
```

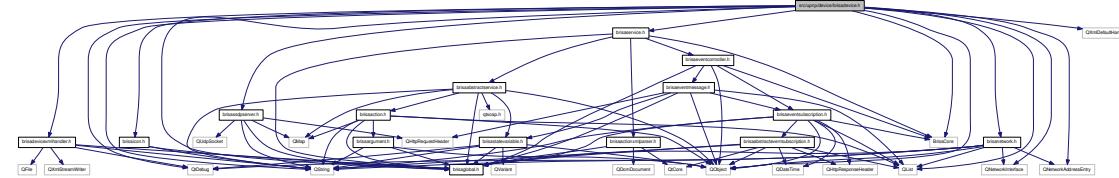
Include dependency graph for brisadevice.cpp:



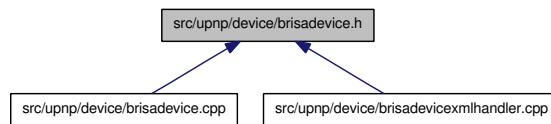
## 7.36 src/upnp/device/brisadevice.h File Reference

```
#include <QString>
#include <QList>
#include <QXmlDefaultHandler>
#include <QNetworkInterface>
#include <QNetworkAddressEntry>
#include <QtDebug>
#include <QObject>
#include <BrisaCore>
#include "brisetwork.h"
#include "brisadevicexmlhandler.h"
#include "brisaservice.h"
#include "brisassdpserver.h"
#include "brisavicon.h"
#include "brisaglobal.h"
```

Include dependency graph for brisadevice.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class `BrisaUpnp::BrisaDevice`  
 $UPnP$  device implementation.

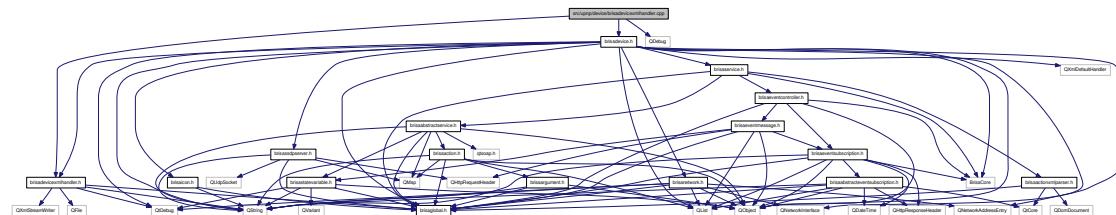
## Namespaces

- namespace BrisaUpnp

## 7.37 src/upnp/device/brisadevicexmlhandler.cpp File Reference

```
#include "brisadevicexmlhandler.h"  
#include "brisadevice.h"  
#include <QDebug>
```

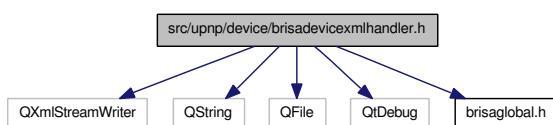
Include dependency graph for brisadevicexmlhandler.cpp:



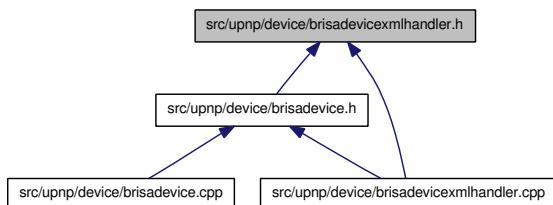
## 7.38 src/upnp/device/brisadevicexmlhandler.h File Reference

```
#include <QXmlStreamWriter>
#include <QString>
#include <QFile>
#include <QtDebug>
#include "brisaglobal.h"
```

Include dependency graph for brisadevicexmlhandler.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaDeviceXMLHandler](#)

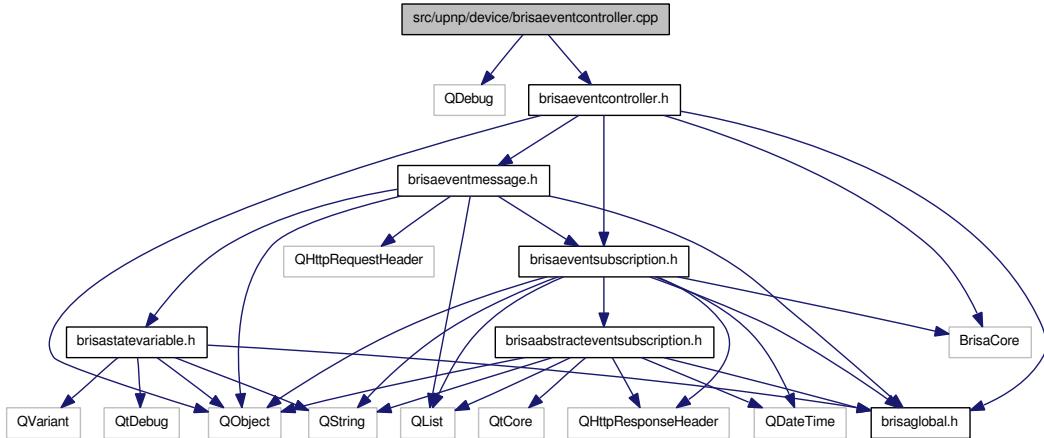
## Namespaces

- namespace [BrisaUpnp](#)

## 7.39 src/upnp/device/brisaeventcontroller.cpp File Reference

```
#include <QDebug>
#include "brisaeventcontroller.h"
```

Include dependency graph for brisaeventcontroller.cpp:



### Defines

- #define [ERROR\\_400\\_MESSAGE](#) "Bad Request"
- #define [ERROR\\_412\\_MESSAGE](#) "Precondition Failed"

#### 7.39.1 Define Documentation

##### 7.39.1.1 #define ERROR\_400\_MESSAGE "Bad Request"

Referenced by BrisaUpnp::BrisaEventController::subscribe(), and BrisaUpnp::BrisaEventController::unsubscribe().

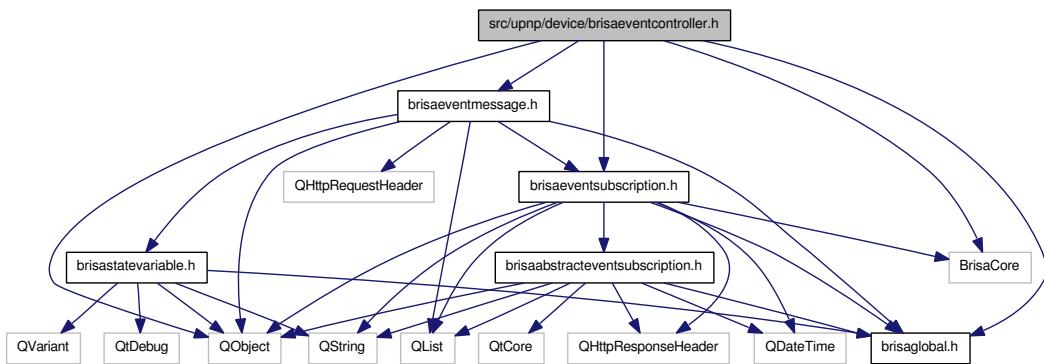
##### 7.39.1.2 #define ERROR\_412\_MESSAGE "Precondition Failed"

Referenced by BrisaUpnp::BrisaEventController::subscribe(), and BrisaUpnp::BrisaEventController::unsubscribe().

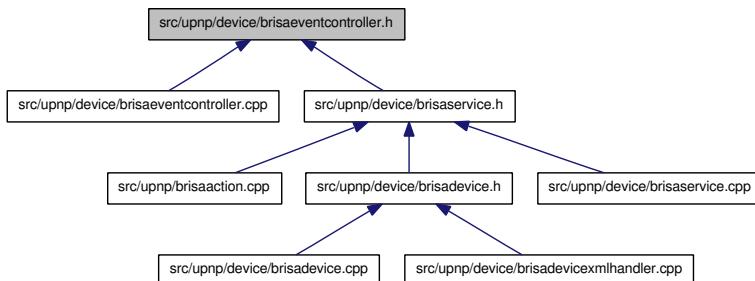
## 7.40 src/upnp/device/brisaeventcontroller.h File Reference

```
#include "brisaeventmessage.h"
#include "brisaeventsSubscription.h"
#include "brisaglobal.h"
#include <BrisaCore>
#include <QObject>
```

Include dependency graph for brisaeventcontroller.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaEventController](#)

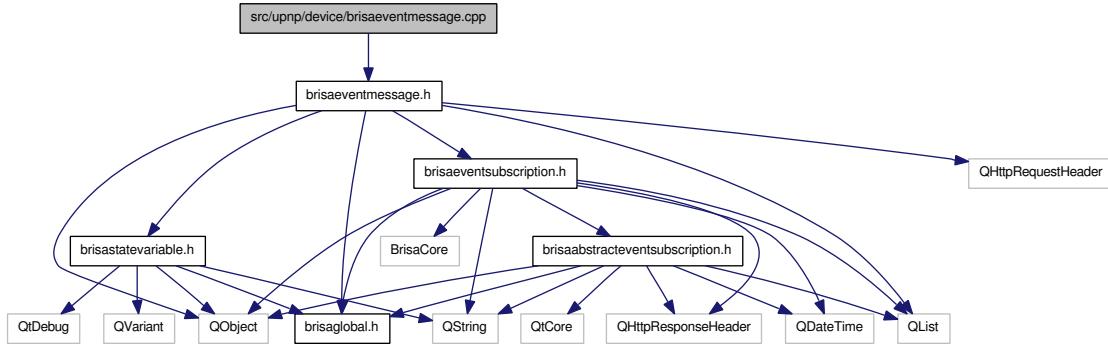
## Namespaces

- namespace [BrisaUpnp](#)

## 7.41 src/upnp/device/brisaeventmessage.cpp File Reference

```
#include "brisaeventmessage.h"
```

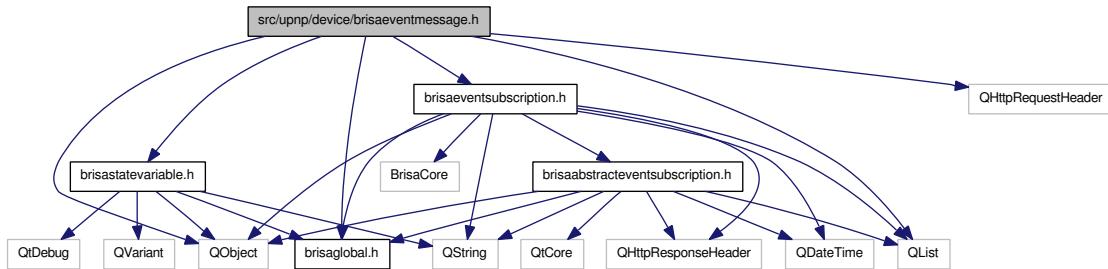
Include dependency graph for brisaeventmessage.cpp:



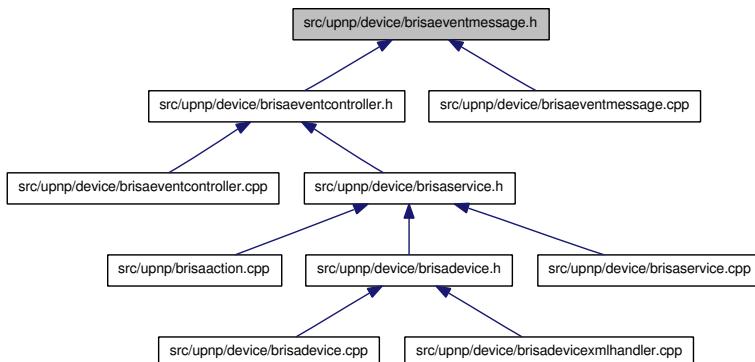
## 7.42 src/upnp/device/brisaeventmessage.h File Reference

```
#include <QObject>
#include <QList>
#include <QHttpRequestHeader>
#include "brisastatevariable.h"
#include "brisaeventsSubscription.h"
#include "brisaglobal.h"
```

Include dependency graph for brisaeventmessage.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaEventMessage](#)

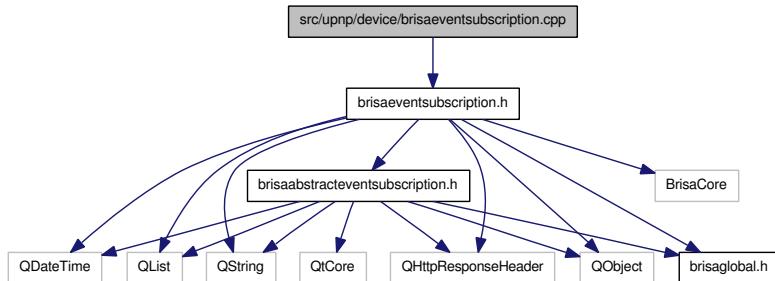
## Namespaces

- namespace [BrisaUpnp](#)

## 7.43 src/upnp/device/brisaeventsubscription.cpp File Reference

```
#include "brisaeventsubscription.h"
```

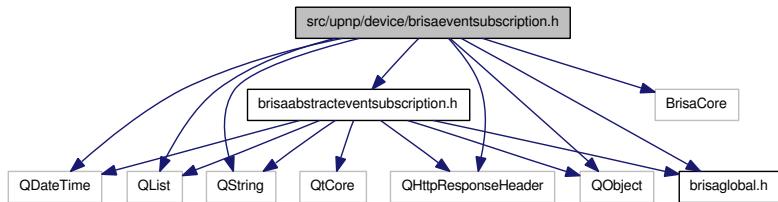
Include dependency graph for brisaeventsubscription.cpp:



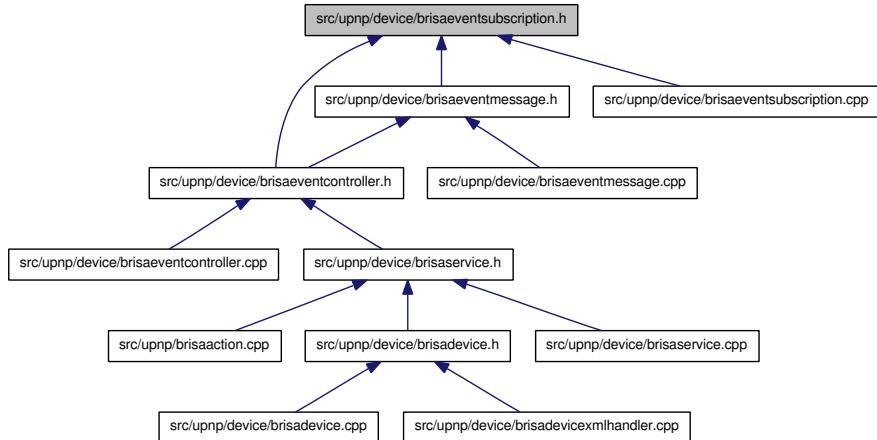
## 7.44 src/upnp/device/brisaeventsubscription.h File Reference

```
#include "brisabSTRACTeventsubscription.h"
#include "brisaglobal.h"
#include <BrisaCore>
#include <QDateTime>
#include <QList>
#include <QString>
#include <QHttpResponseHeader>
#include <QObject>
```

Include dependency graph for brisaeventsubscription.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaEventSubscription](#)

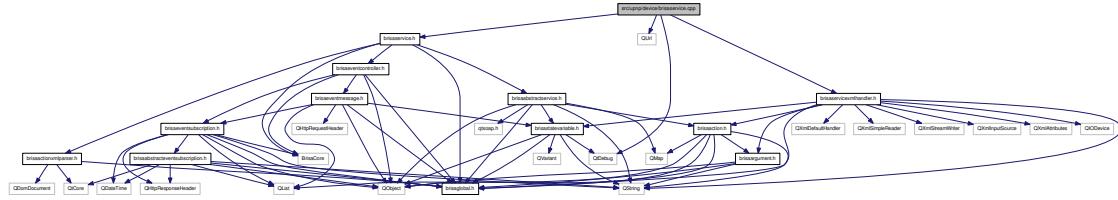
## Namespaces

- namespace [BrisaUpnp](#)

## 7.45 src/upnp/device/brisaservice.cpp File Reference

```
#include <QtDebug>
#include <QUrl>
#include "brisaservice.h"
#include "brisaservicexmlhandler.h"
```

Include dependency graph for brisaservice.cpp:



### Defines

- `#define SOAP_ERROR_TEMPLATE`

#### 7.45.1 Define Documentation

##### 7.45.1.1 #define SOAP\_ERROR\_TEMPLATE

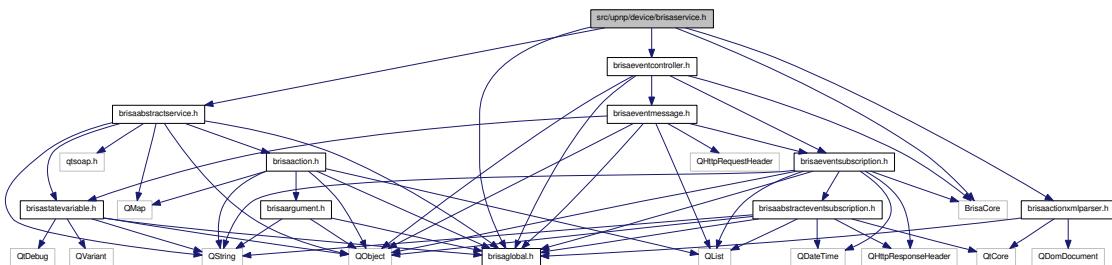
**Value:**

```
"<?xml version=\"1.0\" encoding=\"utf-8\"?>\r\n"
    "<s:Envelope xmlns:s=\"http://schemas.xmlsoap.org/soa
    p/envelope/\" \"\r\n"
        "s:encodingStyle=\"http://schemas.xmlsoap.org/soap/en
        coding/\">\r\n"
            "<s:Body>\r\n"
                "\r\n"
                "<s:Fault>\r\n"
                    "\r\n"
                    "<faultcode>s:Client</faultcode>\r\n"
                    "\r\n"
                    "<faultstring>UPnPError</faultstring>\r\n"
                    "\r\n"
                    "<detail>\r\n"
                        "\r\n"
                        "<UPnPError xmlns=\"urn:schemas-upnp-org:control-1-0\"
                        "\r\n"
                            "<errorCode>%1</errorCode>\r\n"
                            "\r\n"
                            "<errorDescription>%2</errorDescription>\r\n"
                            "\r\n"
                            "</UPnPError>\r\n"
                            "\r\n"
                            "</detail>\r\n"
                            "\r\n"
                            "</s:Fault>\r\n"
                            "\r\n"
                            "</s:Body>\r\n"
                            "\r\n"
                            "</s:Envelope>\r\n"
```

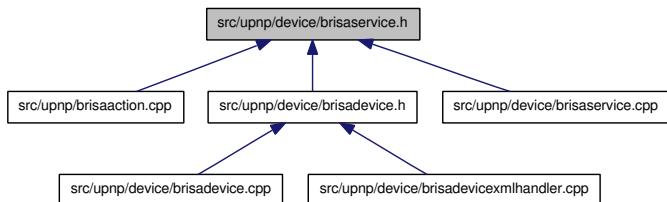
## 7.46 src/upnp/device/brisaservice.h File Reference

```
#include "brisabSTRACTservice.h"
#include "brisaglobal.h"
#include "brisaeVENTcontroller.h"
#include "brisactionxmlparser.h"
#include <BrisaCore>
```

Include dependency graph for brisaservice.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaService](#)  
*UPnP service abstraction.*

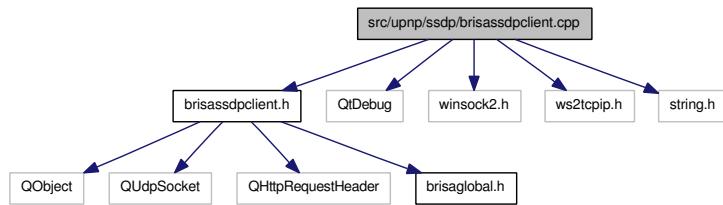
## Namespaces

- namespace [BrisaUpnp](#)

## 7.47 src/upnp/ssdp/brisassdpclient.cpp File Reference

```
#include "brisassdpclient.h"
#include <QtDebug>
#include <winsock2.h>
#include <ws2tcpip.h>
#include <string.h>
```

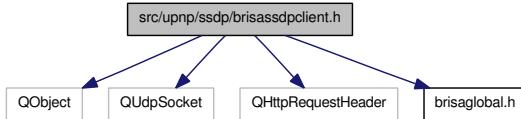
Include dependency graph for brisassdpclient.cpp:



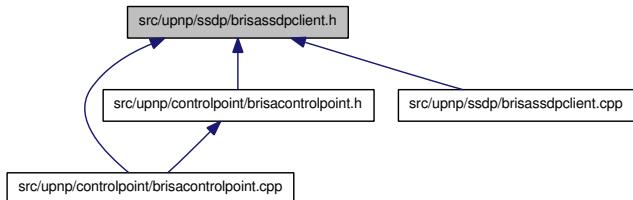
## 7.48 src/upnp/ssdp/brisassdpclient.h File Reference

```
#include <QObject>
#include <QUdpSocket>
#include <QHttpRequestHeader>
#include "brisaglobal.h"
```

Include dependency graph for brisassdpclient.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaSSDPClient](#)  
*SSDP stack implementantion for UPnP control points.*

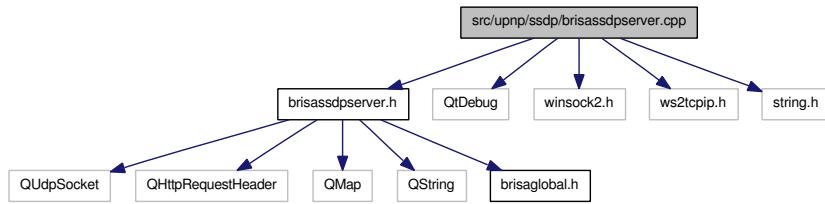
## Namespaces

- namespace [BrisaUpnp](#)

## 7.49 src/upnp/ssdp/brisassdpserver.cpp File Reference

```
#include "brisassdpserver.h"
#include <QtDebug>
#include <winsock2.h>
#include <ws2tcpip.h>
#include <string.h>
```

Include dependency graph for brisassdpserver.cpp:



### Defines

- #define UPNP\_ALIVE\_MESSAGE
- #define UPNP\_BYEBYE\_MESSAGE
- #define UPNP\_MSEARCH\_RESPONSE

#### 7.49.1 Define Documentation

##### 7.49.1.1 #define UPNP\_ALIVE\_MESSAGE

###### Value:

```
"NOTIFY * HTTP/1.1\r\n"          \
    "HOST: 239.255.255.250:1900\r\n" \
    "CACHE-CONTROL: max-age=%1\r\n"   \
    "LOCATION: %2\r\n"               \
    "NT: %3\r\n"                   \
    "NTS: ssdp:alive\r\n"           \
    "SERVER: %4\r\n"                \
    "USN: %5\r\n"                  \
    "\r\n"
```

Referenced by BrisaUpnp::BrisaSSDPServer::doNotify().

##### 7.49.1.2 #define UPNP\_BYEBYE\_MESSAGE

###### Value:

```
"NOTIFY * HTTP/1.1\r\n"          \
    "HOST: 239.255.255.250:1900\r\n" \
    "NT: %1\r\n"                   \
    "NTS: ssdp:byebye\r\n"         \
    "USN: %2\r\n"                  \
    "\r\n"
```

Referenced by BrisaUpnp::BrisaSSDPServer::doByeBye().

#### 7.49.1.3 #define UPNP\_MSEARCH\_RESPONSE

**Value:**

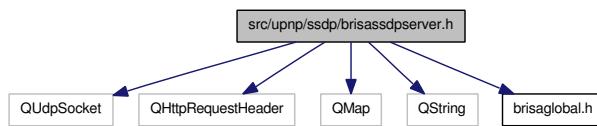
```
"HTTP/1.1 200 OK\r\n"
          \
"CACHE-CONTROL: max-age = %1\r\n" \
"DATE: %2\r\n" \
"EXT: \r\n" \
"LOCATION: %3\r\n" \
"SERVER: %4\r\n" \
"ST: %5\r\n" \
"USN: %6\r\n" \
"\r\n"
```

Referenced by BrisaUpnp::BrisaSSDPServer::respondMSearch().

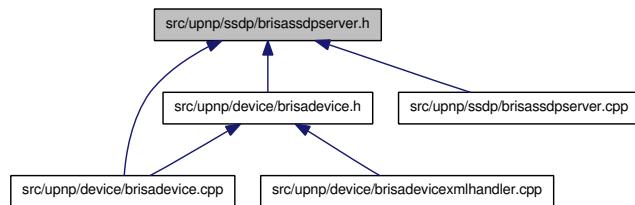
## 7.50 src/upnp/ssdp/brisassdpserver.h File Reference

```
#include <QUdpSocket>
#include <QHttpRequestHeader>
#include <QMap>
#include <QString>
#include "brisaglobal.h"
```

Include dependency graph for brisassdpserver.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [BrisaUpnp::BrisaSSDPServer](#)  
*SSDP stack implementation for UPnP devices.*

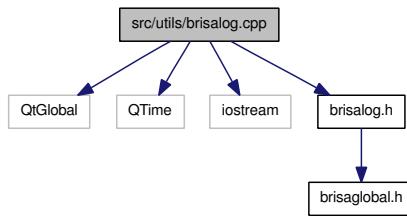
## Namespaces

- namespace [BrisaUpnp](#)

## 7.51 src/utils/brisalog.cpp File Reference

```
#include <QtGlobal>
#include <QTime>
#include <iostream>
#include "brisalog.h"
```

Include dependency graph for brisalog.cpp:



### Defines

- #define COLOR\_DEBUG "\033[32;1m"
- #define COLOR\_WARN "\033[33;1m"
- #define COLOR\_CRITICAL "\033[31;1m"
- #define COLOR\_FATAL "\033[33;1m"
- #define COLOR\_RESET "\033[0m"
- #define LOG\_WRITE(OUTPUT, COLOR, LEVEL, MSG)

### Functions

- static void brisaMessageWriter (QtMsgType type, const char \*msg)
- void brisaLogInitialize (void)

#### 7.51.1 Define Documentation

##### 7.51.1.1 #define COLOR\_CRITICAL "\033[31;1m"

Referenced by brisaMessageWriter().

##### 7.51.1.2 #define COLOR\_DEBUG "\033[32;1m"

Referenced by brisaMessageWriter().

##### 7.51.1.3 #define COLOR\_FATAL "\033[33;1m"

Referenced by brisaMessageWriter().

**7.51.1.4 #define COLOR\_RESET "\033[0m"**

**7.51.1.5 #define COLOR\_WARN "\033[33;1m"**

Referenced by brisaMessageWriter().

**7.51.1.6 #define LOG\_WRITE(OUTPUT, COLOR, LEVEL, MSG)**

**Value:**

```
OUTPUT << COLOR << \
                QDateTime::currentTime().toString("hhmmsszzz").toStdString()
<< \
    " " LEVEL " " << COLOR_RESET << MSG << "\n"
```

Referenced by brisaMessageWriter().

## 7.51.2 Function Documentation

**7.51.2.1 void brisaLogInitialize (void)**

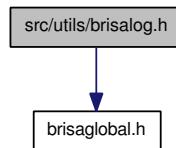
**7.51.2.2 static void brisaMessageWriter (QtMsgType *type*, const char \* *msg*) [static]**

Referenced by brisaLogInitialize().

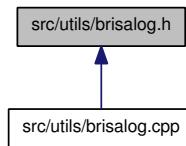
## 7.52 src/utils/brisalog.h File Reference

```
#include "brisaglobal.h"
```

Include dependency graph for brisalog.h:



This graph shows which files directly or indirectly include this file:



### Functions

- BRISA\_UTILS\_EXPORT void [brisalogInitialize](#) (void)

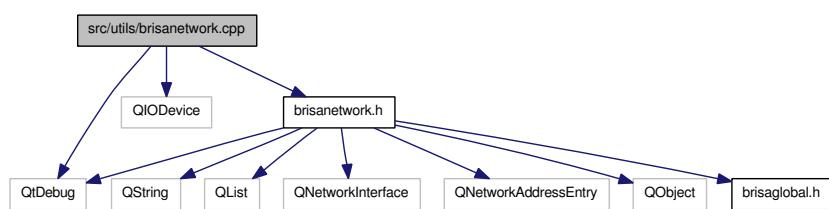
#### 7.52.1 Function Documentation

##### 7.52.1.1 BRISA\_UTILS\_EXPORT void [brisalogInitialize](#) (void)

## 7.53 src/utils/brisanetwork.cpp File Reference

```
#include <QtDebug>
#include <QIODevice>
#include "brisanetwork.h"

Include dependency graph for brisanetwork.cpp:
```



## Functions

- `QString getIp (QString networkInterface)`
- `quint16 getPort ()`

### 7.53.1 Function Documentation

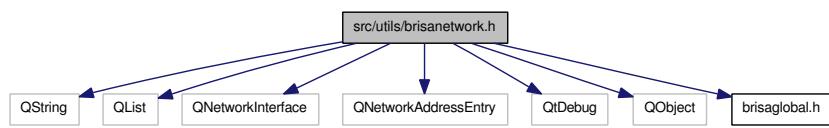
**7.53.1.1 `QString getIp (QString networkInterface)`**

**7.53.1.2 `quint16 getPort ()`**

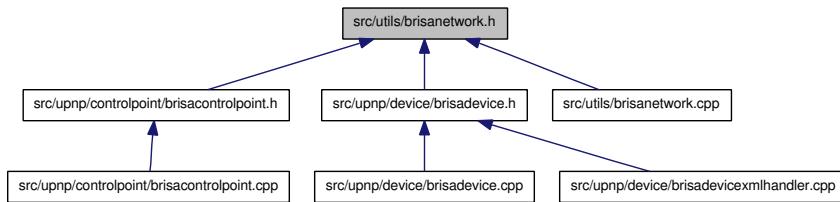
## 7.54 src/utils/brisanetwork.h File Reference

```
#include <QString>
#include <QList>
#include <QNetworkInterface>
#include <QNetworkAddressEntry>
#include <QtDebug>
#include <QObject>
#include "brisaglobal.h"
```

Include dependency graph for brisanetwork.h:



This graph shows which files directly or indirectly include this file:



## Functions

- BRISA\_UTILS\_EXPORT QString [getIp](#) (QString *networkInterface*)
- BRISA\_UTILS\_EXPORT quint16 [getPort](#) ()

### 7.54.1 Function Documentation

#### 7.54.1.1 BRISA\_UTILS\_EXPORT QString getIp (QString *networkInterface*)

#### 7.54.1.2 BRISA\_UTILS\_EXPORT quint16 getPort ()

# Index

~BrisaAbstractService  
    BrisaUpnp::BrisaAbstractService, 35

~BrisaAction  
    BrisaUpnp::BrisaAction, 40

~BrisaActionXmlParser  
    BrisaUpnp::BrisaActionXmlParser, 42

~BrisaControlPoint  
    BrisaUpnp::BrisaControlPoint, 47

~BrisaControlPointDevice  
    BrisaUpnp::BrisaControlPointDevice, 53

~BrisaDevice  
    BrisaUpnp::BrisaDevice, 62

~BrisaEventController  
    BrisaUpnp::BrisaEventController, 77

~BrisaEventProxy  
    BrisaUpnp::BrisaEventProxy, 83

~BrisaMSearchClientCP  
    BrisaUpnp::BrisaMSearchClientCP, 89

~BrisaSSDPClient  
    BrisaUpnp::BrisaSSDPClient, 104

~BrisaSSDPServer  
    BrisaUpnp::BrisaSSDPServer, 107

~BrisaService  
    BrisaUpnp::BrisaService, 93

~BrisaServiceFetcher  
    BrisaUpnp::BrisaServiceFetcher, 96

~BrisaWebFile  
    BrisaCore::BrisaWebFile, 19

~BrisaWebService  
    BrisaCore::BrisaWebService, 24

~BrisaWebServiceProvider  
    BrisaCore::BrisaWebServiceProvider, 27

~BrisaWebStaticContent  
    BrisaCore::BrisaWebStaticContent, 29

~BrisaWebserver  
    BrisaCore::BrisaWebserver, 21

Action  
    BrisaUpnp, 12

ActionList  
    BrisaUpnp, 12

actionList  
    BrisaUpnp::BrisaAbstractService, 36

ActionName  
    BrisaUpnp, 12

addAction  
    BrisaUpnp::BrisaAbstractService, 35

addAllowedValue  
    BrisaUpnp::BrisaStateVariable, 110

addArgument  
    BrisaUpnp::BrisaAction, 40

addArguments  
    BrisaUpnp::BrisaAction, 40

addContent  
    BrisaCore::BrisaWebServiceProvider, 27

addDevice  
    BrisaUpnp::BrisaControlPointDevice, 53

addEmbeddedDevice  
    BrisaUpnp::BrisaDevice, 62

addFile  
    BrisaCore::BrisaWebServiceProvider, 27

addIcon  
    BrisaUpnp::BrisaControlPointDevice, 53

    BrisaUpnp::BrisaDevice, 62

addService  
    BrisaCore::BrisaWebserver, 21

    BrisaUpnp::BrisaControlPointDevice, 53

    BrisaUpnp::BrisaDevice, 62

addStateVariable  
    BrisaUpnp::BrisaAbstractService, 35

AllowedValue  
    BrisaUpnp::BrisaStateVariable, 110

args  
    BrisaUpnp::BrisaActionXmlParser, 43

Argument  
    BrisaUpnp, 12

ArgumentDirection  
    BrisaUpnp, 13

ArgumentList  
    BrisaUpnp, 12

ArgumentName  
    BrisaUpnp, 12

    BrisaUpnp::BrisaArgument, 44

BRISA\_CORE\_EXPORT  
    brisaglobal.h, 116

BRISA\_UPNP\_EXPORT  
    brisaglobal.h, 116

BRISA\_UTILS\_EXPORT  
    brisaglobal.h, 116

BRISA\_VERSION  
brisaglobal.h, 116

BRISA\_VERSION\_STR  
brisaglobal.h, 116

BrisaAbstractEventSubscription  
BrisaUpnp::BrisaAbstractEventSubscription, 31

BrisaAbstractService  
BrisaUpnp::BrisaAbstractService, 35

BrisaAction  
BrisaUpnp::BrisaAction, 40

BrisaActionXmlParser  
BrisaUpnp::BrisaActionXmlParser, 42

BrisaArgument  
BrisaUpnp::BrisaArgument, 44

BrisaConfigurationManager  
BrisaCore::BrisaConfigurationManager, 16

BrisaControlPoint  
BrisaUpnp::BrisaControlPoint, 47  
BrisaUpnp::BrisaEventProxy, 84

BrisaControlPointDevice  
BrisaUpnp::BrisaControlPointDevice, 52

BrisaControlPointService  
BrisaUpnp::BrisaControlPointService, 57

BrisaCore, 9

BrisaCore::BrisaConfigurationManager, 15  
BrisaConfigurationManager, 16

getDirectAccess, 16  
getParameter, 16  
getSectionNames, 17  
items, 17  
removeSection, 17  
save, 17  
setDirectAccess, 17  
setParameter, 18  
update, 18

BrisaCore::BrisaWebFile, 19  
~BrisaWebFile, 19  
BrisaWebFile, 19  
pageRequestedEvent, 19

BrisaCore::BrisaWebserver, 20  
~BrisaWebserver, 21  
addService, 21  
BrisaWebserver, 21  
incomingRequest, 21  
newSession, 21  
publishFile, 21

BrisaCore::BrisaWebService, 23  
~BrisaWebService, 24  
BrisaWebService, 24  
genericRequestReceived, 25  
pageRequestedEvent, 25  
respond, 25

BrisaCore::BrisaWebServiceProvider, 26

~BrisaWebServiceProvider, 27  
addContent, 27  
addFile, 27  
BrisaWebServiceProvider, 27  
indexRequested, 27  
pageRequestedEvent, 27

BrisaCore::BrisaWebStaticContent, 29  
~BrisaWebStaticContent, 29  
BrisaWebStaticContent, 29  
index, 29

BrisaDevice  
BrisaUpnp::BrisaDevice, 61

BrisaDeviceParserContext  
BrisaUpnp::BrisaDeviceParserContext, 68

BrisaEventController  
BrisaUpnp::BrisaEventController, 77

brisaeventcontroller.cpp  
ERROR\_400\_MESSAGE, 154  
ERROR\_412\_MESSAGE, 154

BrisaEventMessage  
BrisaUpnp::BrisaEventMessage, 80

BrisaEventSubscription  
BrisaUpnp::BrisaEventSubscription, 87

brisaglobal.h  
BRISA\_CORE\_EXPORT, 116  
BRISA\_UPNP\_EXPORT, 116  
BRISA\_UTILS\_EXPORT, 116  
BRISA\_VERSION, 116  
BRISA\_VERSION\_STR, 116

BrisaIcon  
BrisaUpnp::BrisaIcon, 88

brisalog.cpp  
brisalogInitialize, 168  
brisamessageWriter, 168  
COLOR\_CRITICAL, 167  
COLOR\_DEBUG, 167  
COLOR\_FATAL, 167  
COLOR\_RESET, 167  
COLOR\_WARN, 168  
LOG\_WRITE, 168

brisalog.h  
brisalogInitialize, 169

brisalogInitialize  
brisalog.cpp, 168  
brisalog.h, 169

brisamessageWriter  
brisalog.cpp, 168

BrisaMSearchClientCP  
BrisaUpnp::BrisaMSearchClientCP, 89

brisamsearchclientcp.cpp  
UPNP\_MSEARCH\_DISCOVER, 146

brisamsearchclientcp.h  
DEFAULT\_SEARCH\_TIME, 147  
DEFAULT\_SEARCH\_TYPE, 147

brisanetwork.cpp  
     getIp, 170  
     getPort, 170  
 brisanetwork.h  
     getIp, 171  
     getPort, 171  
 BrisaService  
     BrisaUpnp::BrisaService, 93  
 brisaservice.cpp  
     SOAP\_ERROR\_TEMPLATE, 160  
 BrisaServiceFetcher  
     BrisaUpnp::BrisaServiceFetcher, 96  
 BrisaServiceParserContext  
     BrisaUpnp::BrisaServiceParserContext, 99  
 BrisaSSDPClient  
     BrisaUpnp::BrisaSSDPClient, 104  
 BrisaSSDPServer  
     BrisaUpnp::BrisaSSDPServer, 107  
 brisassdpservice.cpp  
     UPNP\_ALIVE\_MESSAGE, 164  
     UPNP\_BYE\_BYE\_MESSAGE, 164  
     UPNP\_MSEARCH\_RESPONSE, 164  
 BrisaStateVariable  
     BrisaUpnp::BrisaStateVariable, 110  
 BrisaStateVariableAttribute  
     BrisaUpnp::BrisaStateVariable, 110  
 BrisaUpnp, 10  
     Action, 12  
     ActionList, 12  
     ActionName, 12  
     Argument, 12  
     ArgumentDirection, 13  
     ArgumentList, 12  
     ArgumentName, 12  
     Device, 11  
     DeviceFriendlyName, 11  
     DeviceList, 12  
     DeviceType, 11  
     Error, 12  
     Finished, 12  
     Icon, 12  
     IconDepth, 12  
     IconHeight, 12  
     IconList, 12  
     IconMimetype, 12  
     IconUrl, 12  
     IconWidth, 12  
     Manufacturer, 12  
     ManufacturerUrl, 12  
     ModelDescription, 12  
     modelName, 12  
     ModelUrl, 12  
     PresentationUrl, 12  
     RelatedStateVariable, 13  
     Root, 11  
     SaxParserState, 11  
     saxParserState, 12  
     Scpd, 12  
     SerialNumber, 12  
     Service, 12  
     ServiceControlUrl, 12  
     ServiceError, 13  
     ServiceEventSubUrl, 12  
     ServiceFinished, 13  
     ServiceId, 12  
     ServiceList, 12  
     ServiceScpdUrl, 12  
     ServiceSpecVersion, 12  
     ServiceSpecVersionMajor, 12  
     ServiceSpecVersionMinor, 12  
     ServiceStart, 12  
     ServiceStateTable, 13  
     ServiceType, 12  
     SpecVersion, 11  
     SpecVersionMajor, 11  
     SpecVersionMinor, 11  
     Start, 11  
     StateVariable, 13  
     StateVariableAllowedValue, 13  
     StateVariableAllowedValueList, 13  
     StateVariableAllowedValueRange, 13  
     StateVariableAllowedValueRangeMaximum, 13  
     StateVariableAllowedValueRangeMinimum, 13  
     StateVariableAllowedValueRangeStep, 13  
     StateVariableDataType, 13  
     StateVariableDefaultValue, 13  
     StateVariableName, 13  
     Udn, 12  
     Upc, 12  
     UrlBase, 11  
 BrisaUpnp::BrisaAbstractEventSubscription, 30  
     BrisaAbstractEventSubscription, 31  
     CALLBACK\_URLS, 32  
     date, 32  
     firstMessageSent, 32  
     getCallbackUrls, 31  
     getNextSeq, 31  
     getSid, 31  
     getUrl, 31  
     hasExpired, 32  
     lastSeq, 32  
     renew, 32  
     SID, 32  
     timeout, 32  
 BrisaUpnp::BrisaAbstractService, 33  
     ~BrisaAbstractService, 35

actionList, 36  
addAction, 35  
addStateVariable, 35  
BrisaAbstractService, 35  
call, 35  
clear, 35  
ControlUrl, 35  
controlUrl, 36  
EventSubUrl, 35  
eventSubUrl, 36  
FileAddress, 34  
fileAddress, 36  
getAction, 35  
getActionList, 35  
getAttribute, 36  
getStateVariableList, 36  
Host, 35  
host, 36  
http, 36  
Major, 34  
major, 37  
Minor, 34  
minor, 37  
Port, 35  
port, 37  
requestFinished, 36  
ScpdUrl, 35  
scpdUrl, 37  
ServiceId, 35  
serviceId, 37  
ServiceType, 34  
serviceType, 37  
setAttribute, 36  
stateVariableList, 37  
xmlTags, 34  
BrisaUpnp::BrisaAction, 38  
  ~BrisaAction, 40  
  addArgument, 40  
  addArguments, 40  
  BrisaAction, 40  
  call, 40  
  clearArgumentList, 40  
  getArgumentList, 40  
  getName, 41  
  getService, 41  
  getStateVariable, 41  
  run, 41  
  setName, 41  
  setService, 41  
BrisaUpnp::BrisaActionXmlParser, 42  
  ~BrisaActionXmlParser, 42  
  args, 43  
  BrisaActionXmlParser, 42  
  method, 43  
          parseElement, 43  
          parseSOAP, 43  
          serviceType, 43  
          setXmlContent, 43  
          BrisaUpnp::BrisaArgument, 44  
            ArgumentName, 44  
            BrisaArgument, 44  
            clear, 44  
            Direction, 44  
            getAttribute, 44  
            RelatedStateVariable, 44  
            setAttribute, 44  
            xmlArgument, 44  
          BrisaUpnp::BrisaControlPoint, 45  
            ~BrisaControlPoint, 47  
            BrisaControlPoint, 47  
            deviceFound, 48  
            deviceGone, 48  
            discover, 48  
            getSubscriptionProxy, 48  
            isRunning, 48  
            start, 48  
            stop, 49  
          BrisaUpnp::BrisaControlPointDevice, 50  
            ~BrisaControlPointDevice, 53  
            addDevice, 53  
            addIcon, 53  
            addService, 53  
            BrisaControlPointDevice, 52  
            clear, 53  
            DeviceType, 52  
            FileAddress, 52  
            FriendlyName, 52  
            getAttribute, 53  
            getEmbeddedDeviceList, 53  
            getIconList, 53  
            getServiceById, 54  
            getServiceByType, 54  
            getServiceList, 54  
            Major, 52  
            Manufacturer, 52  
            ManufacturerUrl, 52  
            Minor, 52  
            ModelDescription, 52  
            modelName, 52  
            ModelNumber, 52  
            ModelUrl, 52  
            PresentationUrl, 52  
            SerialNumber, 52  
            setAttribute, 54  
            Udn, 52  
            Upc, 52  
            UrlBase, 52  
            xmlTags, 51

BrisaUpnp::BrisaControlPointService, 55  
     BrisaControlPointService, 57  
     call, 57  
     parseFromXml, 57

BrisaUpnp::BrisaDevice, 58  
     ~BrisaDevice, 62  
     addEmbeddedDevice, 62  
     addIcon, 62  
     addService, 62  
     BrisaDevice, 61  
     clear, 62  
     DeviceType, 61  
     doByeBye, 63  
     doNotify, 63  
     FileAddress, 61  
     FriendlyName, 61  
     getAttribute, 63  
     getEmbeddedDeviceList, 63  
     getIconList, 63  
     getServiceById, 63  
     getServiceByType, 63  
     getServiceList, 64  
     IpAddress, 61  
     Location, 61  
     Major, 61  
     Manufacturer, 61  
     ManufacturerUrl, 61  
     Minor, 61  
     ModelDescription, 61  
     ModelName, 61  
     ModelNumber, 61  
     ModelUrl, 61  
     operator=, 64  
     Port, 61  
     PresentationUrl, 61  
     respondMSearch, 64  
     Running, 61  
     SerialNumber, 61  
     Server, 61  
     setAttribute, 64  
     start, 64  
     stop, 64  
     Udn, 61  
     Upc, 61  
     UrlBase, 61  
     xmlTags, 61

BrisaUpnp::BrisaDeviceParserContext, 67  
     BrisaDeviceParserContext, 68  
     getDevice, 68  
     getIcon, 68  
     getParent, 68  
     getService, 68  
     hasParent, 68  
     setDevice, 69

setIcon, 69  
     setService, 69  
     state, 69  
     stateSkip, 69

BrisaUpnp::BrisaDeviceXMLHandler, 70  
     xmlGenerator, 70

BrisaUpnp::BrisaDeviceXMLHandlerCP, 71  
     characters, 73  
     endElement, 73  
     parseDevice, 73  
     startElement, 73

BrisaUpnp::BrisaEventController, 75  
     ~BrisaEventController, 77  
     BrisaEventController, 77  
     parseGenericRequest, 77  
     subscribe, 77  
     unsubscribe, 77  
     variableChanged, 77

BrisaUpnp::BrisaEventMessage, 79  
     BrisaEventMessage, 80  
     getMessageBody, 80  
     getMessageHeader, 80

BrisaUpnp::BrisaEventProxy, 81  
     ~BrisaEventProxy, 83  
     BrisaControlPoint, 84  
     eventNotification, 83  
     getId, 83  
     renew, 83  
     subscribe, 84  
     unsubscribe, 84

BrisaUpnp::BrisaEventSubscription, 85  
     BrisaEventSubscription, 87  
     getAcceptSubscriptionResponse, 87  
     getAcceptUnsubscriptionResponse, 87  
     renew, 87

BrisaUpnp::BrisaIcon, 88  
     BrisaIcon, 88  
     clear, 88  
     Depth, 88  
     getAttribute, 88  
     Height, 88  
     Mimetype, 88  
     setAttribute, 88  
     Url, 88  
     Width, 88  
     xmlIconTags, 88

BrisaUpnp::BrisaMSearchClientCP, 89  
     ~BrisaMSearchClientCP, 89  
     BrisaMSearchClientCP, 89  
     discover, 89  
     doubleDiscover, 89  
     isRunning, 90  
     msearchResponseReceived, 90  
     start, 90

stop, 90  
BrisaUpnp::BrisaService, 91  
  ~BrisaService, 93  
  BrisaService, 93  
  buildWebServiceTree, 94  
  getVariable, 94  
  getWebService, 94  
  parseGenericRequest, 94  
  setDescriptionFile, 94  
BrisaUpnp::BrisaServiceFetcher, 95  
  ~BrisaServiceFetcher, 96  
  BrisaServiceFetcher, 96  
  fetch, 96  
  fetchFinished, 96  
BrisaUpnp::BrisaServiceParserContext, 98  
  BrisaServiceParserContext, 99  
  getAction, 99  
  getArgument, 99  
  getParent, 99  
  getService, 99  
  getStateVariable, 99  
  hasParent, 100  
  setAction, 100  
  setArgument, 100  
  setService, 100  
  setStateVariable, 100  
  state, 100  
  stateSkip, 100  
BrisaUpnp::BrisaServiceXMLHandler, 101  
  characters, 103  
  endElement, 103  
  parseService, 103  
  startElement, 103  
BrisaUpnp::BrisaSSDPClient, 104  
  ~BrisaSSDPClient, 104  
  BrisaSSDPClient, 104  
  isRunning, 105  
  newDeviceEvent, 105  
  removedDeviceEvent, 105  
  start, 105  
  stop, 105  
BrisaUpnp::BrisaSSDPServer, 106  
  ~BrisaSSDPServer, 107  
  BrisaSSDPServer, 107  
  doByeBye, 107  
  doNotify, 107  
  isRunning, 107  
  msearchRequestReceived, 107  
  respondMSearch, 108  
  start, 108  
  stop, 108  
BrisaUpnp::BrisaStateVariable, 109  
  addAllowedValue, 110  
  AllowedValue, 110  
            BrisaStateVariable, 110  
            BrisaStateVariableAttribute, 110  
            changed, 110  
            clear, 110  
            DataType, 110  
            DefaultValue, 110  
            getAllowedValueList, 111  
            getAttribute, 111  
            getDataType, 111  
            getValue, 111  
            Maximum, 110  
            Minimum, 110  
            Name, 110  
            operator=, 111  
            SendEvents, 110  
            sendEvents, 111  
            setAttribute, 111  
            Step, 110  
            Value, 110  
BrisaWebFile  
  BrisaCore::BrisaWebFile, 19  
BrisaWebserver  
  BrisaCore::BrisaWebserver, 21  
brisawebserver.cpp  
  extractPathLevel, 117  
brisawebserver.h  
  DEFAULT\_PAGE, 119  
BrisaWebService  
  BrisaCore::BrisaWebService, 24  
BrisaWebServiceProvider  
  BrisaCore::BrisaWebServiceProvider, 27  
BrisaWebStaticContent  
  BrisaCore::BrisaWebStaticContent, 29  
buildWebServiceTree  
  BrisaUpnp::BrisaService, 94  
call  
  BrisaUpnp::BrisaAbstractService, 35  
  BrisaUpnp::BrisaAction, 40  
  BrisaUpnp::BrisaControlPointService, 57  
CALLBACK\_URLS  
  BrisaUpnp::BrisaAbstractEventSubscription,  
    32  
changed  
  BrisaUpnp::BrisaStateVariable, 110  
characters  
  BrisaUpnp::BrisaDeviceXMLHandlerCP, 73  
  BrisaUpnp::BrisaServiceXMLHandler, 103  
clear  
  BrisaUpnp::BrisaAbstractService, 35  
  BrisaUpnp::BrisaArgument, 44  
  BrisaUpnp::BrisaControlPointDevice, 53  
  BrisaUpnp::BrisaDevice, 62  
  BrisaUpnp::BrisaIcon, 88

BrisaUpnp::BrisaStateVariable, 110  
 clearArgumentList  
     BrisaUpnp::BrisaAction, 40  
 COLOR\_CRITICAL  
     brisalog.cpp, 167  
 COLOR\_DEBUG  
     brisalog.cpp, 167  
 COLOR\_FATAL  
     brisalog.cpp, 167  
 COLOR\_RESET  
     brisalog.cpp, 167  
 COLOR\_WARN  
     brisalog.cpp, 168  
 ControlUrl  
     BrisaUpnp::BrisaAbstractService, 35  
 controlUrl  
     BrisaUpnp::BrisaAbstractService, 36  
 DataType  
     BrisaUpnp::BrisaStateVariable, 110  
 date  
     BrisaUpnp::BrisaAbstractEventSubscription,  
         32  
 DEFAULT\_PAGE  
     brisawebserver.h, 119  
 DEFAULT\_SEARCH\_TIME  
     brisamsearchclientcp.h, 147  
 DEFAULT\_SEARCH\_TYPE  
     brisamsearchclientcp.h, 147  
 DefaultValue  
     BrisaUpnp::BrisaStateVariable, 110  
 Depth  
     BrisaUpnp::BrisaIcon, 88  
 Device  
     BrisaUpnp, 11  
 deviceFound  
     BrisaUpnp::BrisaControlPoint, 48  
 DeviceFriendlyName  
     BrisaUpnp, 11  
 deviceGone  
     BrisaUpnp::BrisaControlPoint, 48  
 DeviceList  
     BrisaUpnp, 12  
 DeviceType  
     BrisaUpnp, 11  
     BrisaUpnp::BrisaControlPointDevice, 52  
     BrisaUpnp::BrisaDevice, 61  
 Direction  
     BrisaUpnp::BrisaArgument, 44  
 discover  
     BrisaUpnp::BrisaControlPoint, 48  
     BrisaUpnp::BrisaMSearchClientCP, 89  
 doByeBye  
     BrisaUpnp::BrisaDevice, 63  
     BrisaUpnp::BrisaSSDPServer, 107  
 doNotify  
     BrisaUpnp::BrisaDevice, 63  
     BrisaUpnp::BrisaSSDPServer, 107  
 doubleDiscover  
     BrisaUpnp::BrisaMSearchClientCP, 89  
 endElement  
     BrisaUpnp::BrisaDeviceXMLHandlerCP, 73  
     BrisaUpnp::BrisaServiceXMLHandler, 103  
 Error  
     BrisaUpnp, 12  
 ERROR\_400\_MESSAGE  
     brisaeventcontroller.cpp, 154  
 ERROR\_412\_MESSAGE  
     brisaeventcontroller.cpp, 154  
 eventNotification  
     BrisaUpnp::BrisaEventProxy, 83  
 EventSubUrl  
     BrisaUpnp::BrisaAbstractService, 35  
 eventSubUrl  
     BrisaUpnp::BrisaAbstractService, 36  
 extractPathLevel  
     brisawebserver.cpp, 117  
 fetch  
     BrisaUpnp::BrisaServiceFetcher, 96  
 fetchFinished  
     BrisaUpnp::BrisaServiceFetcher, 96  
 FileAddress  
     BrisaUpnp::BrisaAbstractService, 34  
     BrisaUpnp::BrisaControlPointDevice, 52  
     BrisaUpnp::BrisaDevice, 61  
 fileAddress  
     BrisaUpnp::BrisaAbstractService, 36  
 Finished  
     BrisaUpnp, 12  
 firstMessageSent  
     BrisaUpnp::BrisaAbstractEventSubscription,  
         32  
 FriendlyName  
     BrisaUpnp::BrisaControlPointDevice, 52  
     BrisaUpnp::BrisaDevice, 61  
 genericRequestReceived  
     BrisaCore::BrisaWebService, 25  
 getAcceptSubscriptionResponse  
     BrisaUpnp::BrisaEventSubscription, 87  
 getAcceptUnsubscriptionResponse  
     BrisaUpnp::BrisaEventSubscription, 87  
 getAction  
     BrisaUpnp::BrisaAbstractService, 35  
     BrisaUpnp::BrisaServiceParserContext, 99  
 getActionList

BrisaUpnp::BrisaAbstractService, 35  
getAllowedValueList  
    BrisaUpnp::BrisaStateVariable, 111  
getArgument  
    BrisaUpnp::BrisaServiceParserContext, 99  
getArgumentList  
    BrisaUpnp::BrisaAction, 40  
getAttribute  
    BrisaUpnp::BrisaAbstractService, 36  
    BrisaUpnp::BrisaArgument, 44  
    BrisaUpnp::BrisaControlPointDevice, 53  
    BrisaUpnp::BrisaDevice, 63  
    BrisaUpnp::BrisaIcon, 88  
    BrisaUpnp::BrisaStateVariable, 111  
getCallbackUrls  
    BrisaUpnp::BrisaAbstractEventSubscription, 31  
getDataType  
    BrisaUpnp::BrisaStateVariable, 111  
getDevice  
    BrisaUpnp::BrisaDeviceParserContext, 68  
getDirectAccess  
    BrisaCore::BrisaConfigurationManager, 16  
getEmbeddedDeviceList  
    BrisaUpnp::BrisaControlPointDevice, 53  
    BrisaUpnp::BrisaDevice, 63  
getIcon  
    BrisaUpnp::BrisaDeviceParserContext, 68  
getIconList  
    BrisaUpnp::BrisaControlPointDevice, 53  
    BrisaUpnp::BrisaDevice, 63  
getId  
    BrisaUpnp::BrisaEventProxy, 83  
getIp  
    brisanother.cpp, 170  
    brisanother.h, 171  
getMessageBody  
    BrisaUpnp::BrisaEventMessage, 80  
getMessageHeader  
    BrisaUpnp::BrisaEventMessage, 80  
getName  
    BrisaUpnp::BrisaAction, 41  
getNextSeq  
    BrisaUpnp::BrisaAbstractEventSubscription, 31  
getParameter  
    BrisaCore::BrisaConfigurationManager, 16  
getParent  
    BrisaUpnp::BrisaDeviceParserContext, 68  
    BrisaUpnp::BrisaServiceParserContext, 99  
getPort  
    brisanother.cpp, 170  
    brisanother.h, 171  
getSectionNames

    BrisaCore::BrisaConfigurationManager, 17  
getService  
    BrisaUpnp::BrisaAction, 41  
    BrisaUpnp::BrisaDeviceParserContext, 68  
    BrisaUpnp::BrisaServiceParserContext, 99  
getServiceById  
    BrisaUpnp::BrisaControlPointDevice, 54  
    BrisaUpnp::BrisaDevice, 63  
getServiceByType  
    BrisaUpnp::BrisaControlPointDevice, 54  
    BrisaUpnp::BrisaDevice, 63  
getServiceList  
    BrisaUpnp::BrisaControlPointDevice, 54  
    BrisaUpnp::BrisaDevice, 64  
getSid  
    BrisaUpnp::BrisaAbstractEventSubscription, 31  
getStateVariable  
    BrisaUpnp::BrisaAction, 41  
    BrisaUpnp::BrisaServiceParserContext, 99  
getStateVariableList  
    BrisaUpnp::BrisaAbstractService, 36  
getSubscriptionProxy  
    BrisaUpnp::BrisaControlPoint, 48  
getUrl  
    BrisaUpnp::BrisaAbstractEventSubscription, 31  
getValue  
    BrisaUpnp::BrisaStateVariable, 111  
getVariable  
    BrisaUpnp::BrisaService, 94  
getWebService  
    BrisaUpnp::BrisaService, 94  
hasExpired  
    BrisaUpnp::BrisaAbstractEventSubscription, 32  
hasParent  
    BrisaUpnp::BrisaDeviceParserContext, 68  
    BrisaUpnp::BrisaServiceParserContext, 100  
Height  
    BrisaUpnp::BrisaIcon, 88  
Host  
    BrisaUpnp::BrisaAbstractService, 35  
host  
    BrisaUpnp::BrisaAbstractService, 36  
http  
    BrisaUpnp::BrisaAbstractService, 36  
Icon  
    BrisaUpnp, 12  
IconDepth  
    BrisaUpnp, 12  
IconHeight

BrisaUpnp, 12  
 IconList  
     BrisaUpnp, 12  
 IconMimetype  
     BrisaUpnp, 12  
 IconUrl  
     BrisaUpnp, 12  
 IconWidth  
     BrisaUpnp, 12  
 incomingRequest  
     BrisaCore::BrisaWebserver, 21  
 index  
     BrisaCore::BrisaWebStaticContent, 29  
 indexRequested  
     BrisaCore::BrisaWebServiceProvider, 27  
 IpAddress  
     BrisaUpnp::BrisaDevice, 61  
 isRunning  
     BrisaUpnp::BrisaControlPoint, 48  
     BrisaUpnp::BrisaMSearchClientCP, 90  
     BrisaUpnp::BrisaSSDPClient, 105  
     BrisaUpnp::BrisaSSDPServer, 107  
 items  
     BrisaCore::BrisaConfigurationManager, 17  
  
 lastSeq  
     BrisaUpnp::BrisaAbstractEventSubscription,  
         32  
 Location  
     BrisaUpnp::BrisaDevice, 61  
 LOG\_WRITE  
     brisalog.cpp, 168  
  
 Major  
     BrisaUpnp::BrisaAbstractService, 34  
     BrisaUpnp::BrisaControlPointDevice, 52  
     BrisaUpnp::BrisaDevice, 61  
 major  
     BrisaUpnp::BrisaAbstractService, 37  
 Manufacturer  
     BrisaUpnp, 12  
     BrisaUpnp::BrisaControlPointDevice, 52  
     BrisaUpnp::BrisaDevice, 61  
 ManufacturerUrl  
     BrisaUpnp, 12  
     BrisaUpnp::BrisaControlPointDevice, 52  
     BrisaUpnp::BrisaDevice, 61  
 Maximum  
     BrisaUpnp::BrisaStateVariable, 110  
 method  
     BrisaUpnp::BrisaActionXmlParser, 43  
 Mimetype  
     BrisaUpnp::BrisaIcon, 88  
 Minimum

BrisaUpnp::BrisaStateVariable, 110  
 Minor  
     BrisaUpnp::BrisaAbstractService, 34  
     BrisaUpnp::BrisaControlPointDevice, 52  
     BrisaUpnp::BrisaDevice, 61  
 minor  
     BrisaUpnp::BrisaAbstractService, 37  
 ModelDescription  
     BrisaUpnp, 12  
     BrisaUpnp::BrisaControlPointDevice, 52  
     BrisaUpnp::BrisaDevice, 61  
 ModelName  
     BrisaUpnp, 12  
     BrisaUpnp::BrisaControlPointDevice, 52  
     BrisaUpnp::BrisaDevice, 61  
 ModelNumber  
     BrisaUpnp::BrisaControlPointDevice, 52  
     BrisaUpnp::BrisaDevice, 61  
 ModelUrl  
     BrisaUpnp, 12  
     BrisaUpnp::BrisaControlPointDevice, 52  
     BrisaUpnp::BrisaDevice, 61  
 msearchRequestReceived  
     BrisaUpnp::BrisaSSDPServer, 107  
 msearchResponseReceived  
     BrisaUpnp::BrisaMSearchClientCP, 90  
  
 Name  
     BrisaUpnp::BrisaStateVariable, 110  
 newDeviceEvent  
     BrisaUpnp::BrisaSSDPClient, 105  
 newSession  
     BrisaCore::BrisaWebserver, 21  
  
 operator=  
     BrisaUpnp::BrisaDevice, 64  
     BrisaUpnp::BrisaStateVariable, 111  
  
 pageRequestedEvent  
     BrisaCore::BrisaWebFile, 19  
     BrisaCore::BrisaWebService, 25  
     BrisaCore::BrisaWebServiceProvider, 27  
 parseDevice  
     BrisaUpnp::BrisaDeviceXMLHandlerCP, 73  
 parseElement  
     BrisaUpnp::BrisaActionXmlParser, 43  
 parseFromXml  
     BrisaUpnp::BrisaControlPointService, 57  
 parseGenericRequest  
     BrisaUpnp::BrisaEventController, 77  
     BrisaUpnp::BrisaService, 94  
 parseService  
     BrisaUpnp::BrisaServiceXMLHandler, 103  
 parseSOAP

BrisaUpnp::BrisaActionXmlParser, 43  
Port  
    BrisaUpnp::BrisaAbstractService, 35  
    BrisaUpnp::BrisaDevice, 61  
port  
    BrisaUpnp::BrisaAbstractService, 37  
PresentationUrl  
    BrisaUpnp, 12  
    BrisaUpnp::BrisaControlPointDevice, 52  
    BrisaUpnp::BrisaDevice, 61  
publishFile  
    BrisaCore::BrisaWebserver, 21  
  
RelatedStateVariable  
    BrisaUpnp, 13  
    BrisaUpnp::BrisaArgument, 44  
removedDeviceEvent  
    BrisaUpnp::BrisaSSDPClient, 105  
removeSection  
    BrisaCore::BrisaConfigurationManager, 17  
renew  
    BrisaUpnp::BrisaAbstractEventSubscription,  
        32  
    BrisaUpnp::BrisaEventProxy, 83  
    BrisaUpnp::BrisaEventSubscription, 87  
requestFinished  
    BrisaUpnp::BrisaAbstractService, 36  
respond  
    BrisaCore::BrisaWebService, 25  
respondMSearch  
    BrisaUpnp::BrisaDevice, 64  
    BrisaUpnp::BrisaSSDPServer, 108  
Root  
    BrisaUpnp, 11  
run  
    BrisaUpnp::BrisaAction, 41  
Running  
    BrisaUpnp::BrisaDevice, 61  
  
save  
    BrisaCore::BrisaConfigurationManager, 17  
SaxParserState  
    BrisaUpnp, 11  
saxParserState  
    BrisaUpnp, 12  
Scpd  
    BrisaUpnp, 12  
ScpdUrl  
    BrisaUpnp::BrisaAbstractService, 35  
scpdUrl  
    BrisaUpnp::BrisaAbstractService, 37  
SendEvents  
    BrisaUpnp::BrisaStateVariable, 110  
sendEvents  
    BrisaUpnp::BrisaStateVariable, 111  
SerialNumber  
    BrisaUpnp, 12  
    BrisaUpnp::BrisaControlPointDevice, 52  
    BrisaUpnp::BrisaDevice, 61  
Server  
    BrisaUpnp::BrisaDevice, 61  
Service  
    BrisaUpnp, 12  
ServiceControlUrl  
    BrisaUpnp, 12  
ServiceError  
    BrisaUpnp, 13  
ServiceEventSubUrl  
    BrisaUpnp, 12  
ServiceFinished  
    BrisaUpnp, 13  
ServiceId  
    BrisaUpnp, 12  
    BrisaUpnp::BrisaAbstractService, 35  
serviceId  
    BrisaUpnp::BrisaAbstractService, 37  
ServiceList  
    BrisaUpnp, 12  
ServiceScpdUrl  
    BrisaUpnp, 12  
ServiceSpecVersion  
    BrisaUpnp, 12  
ServiceSpecVersionMajor  
    BrisaUpnp, 12  
ServiceSpecVersionMinor  
    BrisaUpnp, 12  
ServiceStart  
    BrisaUpnp, 12  
ServiceStateTable  
    BrisaUpnp, 13  
ServiceType  
    BrisaUpnp, 12  
    BrisaUpnp::BrisaAbstractService, 34  
serviceType  
    BrisaUpnp::BrisaAbstractService, 37  
    BrisaUpnp::BrisaActionXmlParser, 43  
setAction  
    BrisaUpnp::BrisaServiceParserContext, 100  
setArgument  
    BrisaUpnp::BrisaServiceParserContext, 100  
setAttribute  
    BrisaUpnp::BrisaAbstractService, 36  
    BrisaUpnp::BrisaArgument, 44  
    BrisaUpnp::BrisaControlPointDevice, 54  
    BrisaUpnp::BrisaDevice, 64  
    BrisaUpnp::BrisaIcon, 88  
    BrisaUpnp::BrisaStateVariable, 111  
setDescriptionFile

BrisaUpnp::BrisaService, 94  
 setDevice  
     BrisaUpnp::BrisaDeviceParserContext, 69  
 setDirectAccess  
     BrisaCore::BrisaConfigurationManager, 17  
 setIcon  
     BrisaUpnp::BrisaDeviceParserContext, 69  
 setName  
     BrisaUpnp::BrisaAction, 41  
 setParameter  
     BrisaCore::BrisaConfigurationManager, 18  
 setService  
     BrisaUpnp::BrisaAction, 41  
     BrisaUpnp::BrisaDeviceParserContext, 69  
     BrisaUpnp::BrisaServiceParserContext, 100  
 setStateVariable  
     BrisaUpnp::BrisaServiceParserContext, 100  
 setXmlContent  
     BrisaUpnp::BrisaActionXmlParser, 43  
 SID  
     BrisaUpnp::BrisaAbstractEventSubscription, 32  
 SOAP\_ERROR\_TEMPLATE  
     brisaservice.cpp, 160  
 SpecVersion  
     BrisaUpnp, 11  
 SpecVersionMajor  
     BrisaUpnp, 11  
 SpecVersionMinor  
     BrisaUpnp, 11  
 src/core/brisacconfig.cpp, 113  
 src/core/brisacconfig.h, 114  
 src/core/brisacore.h, 115  
 src/core/brisaglobal.h, 116  
 src/core/brisawebserver.cpp, 117  
 src/core/brisawebserver.h, 118  
 src/upnp/brisabSTRACTeventsSubscription.cpp, 120  
 src/upnp/brisabSTRACTeventsSubscription.h, 121  
 src/upnp/brisabSTRACTservice.cpp, 122  
 src/upnp/brisabSTRACTservice.h, 123  
 src/upnp/brisaction.cpp, 124  
 src/upnp/brisaction.h, 125  
 src/upnp/brisargument.cpp, 126  
 src/upnp/brisargument.h, 127  
 src/upnp/brisavicon.cpp, 128  
 src/upnp/brisavicon.h, 129  
 src/upnp/brisaserviceXMLhandler.cpp, 130  
 src/upnp/brisaserviceXMLhandler.h, 131  
 src/upnp/brisastatevariable.cpp, 133  
 src/upnp/brisastatevariable.h, 134  
 src/upnp/controlpoint/brisaccontrolpoint.cpp, 135  
 src/upnp/controlpoint/brisaccontrolpoint.h, 136  
 src/upnp/controlpoint/brisaccontrolpointdevice.cpp, 137  
 src/upnp/controlpoint/brisaccontrolpointdevice.h, 138  
 src/upnp/controlpoint/brisaccontrolpointservice.cpp, 139  
 src/upnp/controlpoint/brisaccontrolpointservice.h, 140  
 src/upnp/controlpoint/brisadeviceXMLhandler.cpp, 141  
 src/upnp/controlpoint/brisadeviceXMLhandler.h, 142  
 src/upnp/controlpoint/brisaeventproxy.cpp, 144  
 src/upnp/controlpoint/brisaeventproxy.h, 145  
 src/upnp/controlpoint/brisamsearchclientcp.cpp, 146  
 src/upnp/controlpoint/brisamsearchclientcp.h, 147  
 src/upnp/device/brisactionXMLparser.cpp, 148  
 src/upnp/device/brisactionXMLparser.h, 149  
 src/upnp/device/brisadevice.cpp, 150  
 src/upnp/device/brisadevice.h, 151  
 src/upnp/device/brisadeviceXMLhandler.cpp, 152  
 src/upnp/device/brisadeviceXMLhandler.h, 153  
 src/upnp/device/brisaeventcontroller.cpp, 154  
 src/upnp/device/brisaeventcontroller.h, 155  
 src/upnp/device/brisaeventmessage.cpp, 156  
 src/upnp/device/brisaeventmessage.h, 157  
 src/upnp/device/brisaeventsSubscription.cpp, 158  
 src/upnp/device/brisaeventsSubscription.h, 159  
 src/upnp/device/brisaservice.cpp, 160  
 src/upnp/device/brisaservice.h, 161  
 src/upnp/ssdp/brisassdpclient.cpp, 162  
 src/upnp/ssdp/brisassdpclient.h, 163  
 src/upnp/ssdp/brisassdpserver.cpp, 164  
 src/upnp/ssdp/brisassdpserver.h, 166  
 src/utils/brisalog.cpp, 167  
 src/utils/brisalog.h, 169  
 src/utils/brisanetwork.cpp, 170  
 src/utils/brisanetwork.h, 171  
 Start  
     BrisaUpnp, 11  
 start  
     BrisaUpnp::BrisaControlPoint, 48  
     BrisaUpnp::BrisaDevice, 64  
     BrisaUpnp::BrisaMSearchClientCP, 90  
     BrisaUpnp::BrisaSSDPClient, 105  
     BrisaUpnp::BrisaSSDPServer, 108  
 startElement  
     BrisaUpnp::BrisaDeviceXMLHandlerCP, 73  
     BrisaUpnp::BrisaServiceXMLHandler, 103  
 state  
     BrisaUpnp::BrisaDeviceParserContext, 69  
     BrisaUpnp::BrisaServiceParserContext, 100  
 stateSkip  
     BrisaUpnp::BrisaDeviceParserContext, 69  
     BrisaUpnp::BrisaServiceParserContext, 100

StateVariable  
    BrisaUpnp, 13

StateVariableAllowedValue  
    BrisaUpnp, 13

StateVariableAllowedValueList  
    BrisaUpnp, 13

StateVariableAllowedValueRange  
    BrisaUpnp, 13

StateVariableAllowedValueRangeMaximum  
    BrisaUpnp, 13

StateVariableAllowedValueRangeMinimum  
    BrisaUpnp, 13

StateVariableAllowedValueRangeStep  
    BrisaUpnp, 13

StateVariableDataType  
    BrisaUpnp, 13

StateVariableDefaultValue  
    BrisaUpnp, 13

stateVariableList  
    BrisaUpnp::BrisaAbstractService, 37

StateVariableName  
    BrisaUpnp, 13

Step  
    BrisaUpnp::BrisaStateVariable, 110

stop  
    BrisaUpnp::BrisaControlPoint, 49  
    BrisaUpnp::BrisaDevice, 64  
    BrisaUpnp::BrisaMSearchClientCP, 90  
    BrisaUpnp::BrisaSSDPCClient, 105  
    BrisaUpnp::BrisaSSDPServer, 108

subscribe  
    BrisaUpnp::BrisaEventController, 77  
    BrisaUpnp::BrisaEventProxy, 84

timeout  
    BrisaUpnp::BrisaAbstractEventSubscription,  
        32

Udn  
    BrisaUpnp, 12  
    BrisaUpnp::BrisaControlPointDevice, 52  
    BrisaUpnp::BrisaDevice, 61

unsubscribe  
    BrisaUpnp::BrisaEventController, 77  
    BrisaUpnp::BrisaEventProxy, 84

Upc  
    BrisaUpnp, 12  
    BrisaUpnp::BrisaControlPointDevice, 52  
    BrisaUpnp::BrisaDevice, 61

update  
    BrisaCore::BrisaConfigurationManager, 18

UPNP\_ALIVE\_MESSAGE  
    brisassdpservice.cpp, 164

UPNP\_BYE\_BYE\_MESSAGE