Ev3Dev

0.1.1

Generated by Doxygen 1.8.11

Contents

1 Hierarchical Index **1**

1.1 Class Hierarchy 1

2 Class Index **3**

2.1 Class List 3

3 File Index **7**

3.1 File List 7

4 Class Documentation **9**

4.1 ev3::Action Class Reference 9

4.1.1 Detailed Description 10

4.1.2 Member Enumeration Documentation 11

4.1.2.1 ActionType 11

4.1.3 Constructor & Destructor Documentation 11

4.1.3.1 Action(CommandsVector commands, ActionType type) 11

4.1.3.2 Action(CommandsVector commands) 11

4.1.3.3 Action(ActionType type) 11

4.1.4 Member Function Documentation 12

4.1.4.1 getActionPrototype() 12

4.1.4.2 getString() 12

4.1.4.3 getType() 12

4.1.4.4 isExecuted() 12

4.1.4.5 isFinished() 12

4.1.4.6 setCommands(CommandsVector commands) 12

4.1.4.7 setEndCondition(EndCondition condition) 13
4.1.5 Member Data Documentation 13
4.1.5.1 _endCondition 13
4.2 ev3::ActionDriveDistance Class Reference 13
4.2.1 Detailed Description 14
4.2.2 Constructor & Destructor Documentation 14
4.2.2.1 ActionDriveDistance(int distance) 14
4.2.2.2 ActionDriveDistance(CommandsVector commands, int distance) 14
4.2.3 Member Function Documentation 14
4.2.3.1 getActionPrototype() 14
4.2.3.2 getDistance() 15
4.2.3.3 getString() override 15
4.3 ev3::ActionDriveForever Class Reference 15
4.3.1 Detailed Description 16
4.3.2 Constructor & Destructor Documentation 16
4.3.2.1 ActionDriveForever(bool forward=true) 16
4.3.2.2 ActionDriveForever(CommandsVector commands, bool forward=true) 16
4.3.3 Member Function Documentation 16
4.3.3.1 getActionPrototype() 16
4.3.3.2 getString() override 17
4.3.3.3 isForward() 17
4.4 ev3::ActionRepeat Class Reference 17
4.4.1 Detailed Description 18
4.4.2 Constructor & Destructor Documentation 18
4.4.2.1 ActionRepeat(StoredActions actions, unsigned int n) 18
4.4.3 Member Function Documentation 18
4.4.3.1 getString() 18
4.5 ev3::ActionRotate Class Reference 18
4.5.1 Detailed Description 19
4.5.2 Constructor & Destructor Documentation 19

4.5.2.1 ActionRotate(int rotation) 19

4.5.2.2 ActionRotate(CommandsVector commands, int rotation) 19

4.5.3 Member Function Documentation 20

4.5.3.1 getActionPrototype() 20

4.5.3.2 getRotation() 20

4.5.3.3 getString() override 20

4.6 ev3::ActionRotateRandDirection Class Reference 20

4.6.1 Detailed Description 21

4.6.2 Constructor & Destructor Documentation 21

4.6.2.1 ActionRotateRandDirection(int rotation) 21

4.6.2.2 ActionRotateRandDirection(CommandsVector commands, int rotation) 21

4.6.3 Member Function Documentation 21

4.6.3.1 execute() override 21

4.6.3.2 getActionPrototype() 22

4.6.3.3 getString() override 22

4.7 ev3::ActionStop Class Reference 22

4.7.1 Detailed Description 23

4.7.2 Constructor & Destructor Documentation 23

4.7.2.1 ActionStop(CommandsVector commands) 23

4.7.3 Member Function Documentation 23

4.7.3.1 getActionPrototype() 23

4.7.3.2 getString() override 23

4.8 ev3::Agent Class Reference 24

4.8.1 Detailed Description 24

4.8.2 Member Function Documentation 24

4.8.2.1 getCommId() 24

4.8.2.2 getId() 25

4.8.2.3 processMessage(Message ∗message, Message ∗retMessage) 25

4.8.2.4 setBehaviour(SharedPtrBehaviour behaviour) 25

4.8.2.5 setCommId(const unsigned int commId) 25

4.8.2.6 setId(const unsigned int id) 25
4.8.2.7 setMeasurement(Measurements measurements) 26
4.8.2.8 updateLastMessage(Message ∗message) 26
4.9 ev3::Behaviour Class Reference 26
4.9.1 Detailed Description 28
4.9.2 Member Enumeration Documentation 28
4.9.2.1 BehaviourType 28
4.9.3 Constructor & Destructor Documentation 28
4.9.3.1 Behaviour(BehaviourType type, BehaviourStates states) 28
4.9.3.2 Behaviour(BehaviourType type) 28
4.9.4 Member Function Documentation 28
4.9.4.1 getPrototype() 28
4.9.4.2 getString() 29
4.9.4.3 react(Event::EventType type) 29
4.9.4.4 setMeasurements(Measurements measurements) 29
4.9.4.5 setReactionStates(BehaviourStates reactionStates) 29
4.9.4.6 setStates(BehaviourStates states) 29
4.9.4.7 setStopState(BehaviourState state) 30
4.10 ev3::BehaviourDriveOnSquare Class Reference 30
4.10.1 Detailed Description 30
4.10.2 Constructor & Destructor Documentation 31
4.10.2.1 BehaviourDriveOnSquare(unsigned int side, bool turningRight) 31
4.10.2.2 BehaviourDriveOnSquare(BehaviourStates states, unsigned int side, bool turningRight) 31
4.10.3 Member Function Documentation 31
4.10.3.1 getPrototype() 31
4.10.3.2 getString() 31
4.11 ev3::BehaviourExploreRandom Class Reference 32
4.11.1 Detailed Description 32
4.11.2 Constructor & Destructor Documentation 32
4.11.2.1 BehaviourExploreRandom(BehaviourStates states) 32

4.11.3 Member Function Documentation 32
4.11.3.1 getPrototype() 32
4.11.3.2 getString() 33
4.12 ev3::BehaviourState Class Reference 33
4.12.1 Detailed Description 34
4.12.2 Constructor & Destructor Documentation 34
4.12.2.1 BehaviourState(const BehaviourState &)=default 34
4.12.2.2 BehaviourState(SharedPtrAction action, unsigned int nextState, bool isStop↩ State=false) 34
4.12.2.3 BehaviourState(SharedPtrAction action, unsigned int nextState, Reactions↩ Transitions reactions) 34
4.12.3 Member Function Documentation 34
4.12.3.1 getAction() 34
4.12.3.2 getReaction(Event::EventType type) 35
4.12.3.3 isStopState() 35
4.12.3.4 process() 35
4.12.3.5 setNextState(const unsigned int next) 35
4.12.3.6 setReactions(ReactionsTransitions reactions) 35
4.13 ev3dev::button Class Reference 36
4.14 ev3::CircularBuffer< T > Class Template Reference 36
4.15 ev3dev::color_sensor Class Reference 37
4.16 ev3::ColorUtils Class Reference 37
4.17 ev3::Command Class Reference 38
4.17.1 Detailed Description 39
4.17.2 Member Function Documentation 39
4.17.2.1 getString() 39
4.18 ev3::CommandMotor Class Reference 39
4.18.1 Detailed Description 40
4.18.2 Constructor & Destructor Documentation 40
4.18.2.1 CommandMotor(Motor &motor) 40
4.18.3 Member Function Documentation 40

4.18.3.1 getMotor() 40
4.19 ev3::CommandMotorReset Class Reference 41
4.19.1 Detailed Description 41
4.19.2 Constructor & Destructor Documentation 41
4.19.2.1 CommandMotorReset(Motor &motor) 41
4.20 ev3::CommandMotorRunForever Class Reference 42
4.20.1 Detailed Description 42
4.20.2 Constructor & Destructor Documentation 42
4.20.2.1 CommandMotorRunForever(Motor &motor) 42
4.21 ev3::CommandMotorSetSpeed Class Reference 43
4.21.1 Detailed Description 43
4.21.2 Constructor & Destructor Documentation 43
4.21.2.1 CommandMotorSetSpeed(Motor &motor, int value) 43
4.22 ev3::CommandMotorSetSpeedRegEnabled Class Reference 44
4.22.1 Detailed Description 44
4.22.2 Constructor & Destructor Documentation 44
4.22.2.1 CommandMotorSetSpeedRegEnabled(Motor &motor, bool value) 44
4.23 ev3::CommandMotorSetStopMode Class Reference 45
4.23.1 Detailed Description 45
4.23.2 Member Enumeration Documentation 46
4.23.2.1 StopMode 46
4.23.3 Constructor & Destructor Documentation 46
4.23.3.1 CommandMotorSetStopMode(Motor &motor, StopMode mode) 46
4.24 ev3::CommandMotorStop Class Reference 46
4.24.1 Detailed Description 47
4.24.2 Constructor & Destructor Documentation 47
4.24.2.1 CommandMotorStop(Motor &motor) 47
4.25 ev3::CommandSensor Class Reference 47
4.25.1 Detailed Description 48
4.25.2 Constructor & Destructor Documentation 48

4.25.2.1 CommandSensor(Sensor &sensor) 48
4.25.3 Member Function Documentation 48
4.25.3.1 getSensor() 48
4.26 ev3::Communication Class Reference 48
4.26.1 Detailed Description 49
4.26.2 Member Function Documentation 49
4.26.2.1 createThread(Queue< Message > ∗sendQueue, Queue< Message > ∗receiveQueue, bool isMaster=false) 49
4.26.2.2 run(Queue< Message > ∗sendQueue, Queue< Message > ∗receiveQueue, bool isMaster=false) 49
4.27 ev3::CommUtils Class Reference 50
4.27.1 Detailed Description 50
4.27.2 Member Function Documentation 50
4.27.2.1 preparePassiveSocket(unsigned int portNumber) 50
4.27.2.2 receiveMessage(unsigned int socket, Message &message, NetworkNode &sender) 51
4.27.2.3 receiveMessageDelay(unsigned int socket, Message &message, NetworkNode &sender, unsigned int msDelay=DEFAULT_RECEIVE_DELAY) 51
4.27.2.4 sendMessage(unsigned int socket, unsigned int port, Message &message, std↩ ::string &proto, bool isMaster, unsigned int repeat=SENT_MESSAGE_COPIES) . 51
4.28 ev3dev::dc_motor Class Reference 52
4.29 ev3dev::device Class Reference 53
4.30 ev3::Devices Class Reference 54
4.30.1 Detailed Description 55
4.30.2 Constructor & Destructor Documentation 55
4.30.2.1 Devices(const Devices &other) 55
4.30.3 Member Function Documentation 56
4.30.3.1 addListener(Sensor::SensorType type) 56
4.30.3.2 checkDevices(RequiredDevices &devices) 56
4.30.3.3 getInstance() 56
4.30.3.4 getMotor(ev3dev::port_type port) 56
4.30.3.5 getSensor(ev3dev::port_type port) 57
4.30.3.6 operator=(const Devices &other) 57

4.30.3.7 removeListener(Sensor::SensorType type) 57
4.30.3.8 setProximitySensor(ev3dev::port_type port) 57
4.30.3.9 setSafetyTouchSensor(ev3dev::port_type port) 58
4.31 ev3::Event Class Reference 58
4.31.1 Detailed Description 59
4.31.2 Member Enumeration Documentation 59
4.31.2.1 EventType 59
4.31.3 Constructor & Destructor Documentation 59
4.31.3.1 Event(EventType type) 59
4.31.4 Member Function Documentation 59
4.31.4.1 getStringType() 59
4.31.4.2 getType() 60
4.32 ev3::EventAction Class Reference 60
4.32.1 Detailed Description 60
4.32.2 Constructor & Destructor Documentation 60
4.32.2.1 EventAction(EventType eventType, Action::ActionType actionType) 60
4.32.3 Member Function Documentation 61
4.32.3.1 getActionType() 61
4.33 ev3::EventQueue Class Reference 61
4.34 ev3::EventSensorWatch Class Reference 62
4.34.1 Detailed Description 62
4.34.2 Constructor & Destructor Documentation 62
4.34.2.1 EventSensorWatch(Sensor::SensorType type, SensorValue value) 62
4.34.3 Member Function Documentation 63
4.34.3.1 getType() 63
4.34.3.2 getValue() 63
4.35 ev3dev::gyro_sensor Class Reference 63
4.36 ev3dev::i2c_sensor Class Reference 64
4.37 ev3dev::infrared_sensor Class Reference 65
4.38 ev3dev::large_motor Class Reference 65

4.39 ev3dev::lcd Class Reference 66
4.40 ev3dev::led Class Reference 66
4.41 ev3::LedControl Class Reference 68
4.41.1 Detailed Description 68
4.41.2 Member Enumeration Documentation 69
4.41.2.1 LedColors 69
4.41.2.2 LedType 69
4.41.3 Member Function Documentation 69
4.41.3.1 flash(unsigned int leds, unsigned int msInterval, unsigned int repeat=1, unsigned int brightnessRed=MAX_BRIGHTNESS, unsigned int brightnessGreen=MAX_↩ BRIGHTNESS) 69
4.41.3.2 flashColor(LedColors color, unsigned int msInterval, unsigned int repeat=1) . . . 69
4.41.3.3 off(unsigned int leds=LedType::ALL) 70
4.41.3.4 on(unsigned int leds=LedType::ALL, unsigned int brightness=MAX_BRIGHTN↩ ESS) 70
4.41.3.5 onExclusive(unsigned int leds=LedType::ALL, unsigned int brightness=MAX_B↩ RIGHTNESS) 70
4.41.3.6 setColor(LedColors color) 70
4.42 ev3dev::lego_port Class Reference 71
4.43 ev3dev::light_sensor Class Reference 71
4.44 ev3::Logger Class Reference 72
4.45 ev3::Master Class Reference 73
4.45.1 Detailed Description 73
4.45.2 Member Function Documentation 73
4.45.2.1 createThread(Queue< Message > ∗sendQueue, Queue< Message > ∗receiveQueue) 73
4.45.2.2 run(Queue< Message > ∗sendQueue, Queue< Message > ∗receiveQueue) . 73
4.45.2.3 send(Message message, bool recordMessage=true) 74
4.46 ev3dev::medium_motor Class Reference 74
4.47 ev3::Message Class Reference 75
4.47.1 Detailed Description 76
4.47.2 Member Enumeration Documentation 76
4.47.2.1 MessageType 76

4.47.3 Constructor & Destructor Documentation . . . 76
4.47.3.1 Message(unsigned int senderId, unsigned int receiverId, unsigned int messageId, MessageType type, StringVector parameters={}) . . . 76
4.47.4 Member Function Documentation . . . 77
4.47.4.1 decodeMessage(const std::string message) . . . 77
4.47.4.2 empty() . . . 77
4.47.4.3 encodeMessage(Message &message) . . . 77
4.47.4.4 getMessageId() . . . 78
4.47.4.5 getParameters() . . . 78
4.47.4.6 getReceiverId() . . . 78
4.47.4.7 getSenderId() . . . 78
4.47.4.8 getString() . . . 78
4.47.4.9 getType() . . . 78
4.47.4.10 setMessageId(unsigned int id) . . . 78
4.47.4.11 setParameters(StringVector parameters) . . . 79
4.47.4.12 setReceiverId(unsigned int id) . . . 79
4.47.4.13 setSenderId(unsigned int id) . . . 79
4.47.4.14 setType(MessageType type) . . . 79
4.48 ev3::Motor Class Reference . . . 80
4.48.1 Detailed Description . . . 80
4.48.2 Constructor & Destructor Documentation . . . 80
4.48.2.1 Motor(ev3dev::motor motor) . . . 80
4.48.3 Member Function Documentation . . . 80
4.48.3.1 getMotor() . . . 80
4.49 ev3dev::motor Class Reference . . . 81
4.50 ev3::CommUtils::NetworkNode Struct Reference . . . 83
4.50.1 Detailed Description . . . 83
4.51 ev3dev::power_supply Class Reference . . . 83
4.52 ev3::Queue< T > Class Template Reference . . . 84
4.53 ev3dev::remote_control Class Reference . . . 84
4.54 ev3::Robot Class Reference . . . 85

4.55 ev3::RobotModelA Class Reference . . . 86
4.56 ev3::RobotState Class Reference . . . 86
4.57 ev3::RobotStateActive Class Reference . . . 87
4.58 ev3::RobotStateIdle Class Reference . . . 88
4.59 ev3::RobotStatePanic Class Reference . . . 88
4.60 ev3::RobotStatePaused Class Reference . . . 89
4.61 ev3::RobotStateWorking Class Reference . . . 89
4.62 ev3dev::sensor Class Reference . . . 90
4.63 ev3::Sensor Class Reference . . . 92
4.63.1 Detailed Description . . . 92
4.63.2 Member Enumeration Documentation . . . 92
4.63.2.1 SensorType . . . 92
4.63.3 Constructor & Destructor Documentation . . . 92
4.63.3.1 Sensor(ev3dev::sensor sensor, SensorType type) . . . 92
4.63.4 Member Function Documentation . . . 93
4.63.4.1 getDecimals() . . . 93
4.63.4.2 getNumValues() . . . 93
4.63.4.3 getSensor() . . . 93
4.63.4.4 getType() . . . 93
4.63.4.5 getValue(unsigned int n) . . . 93
4.63.4.6 getValueF(unsigned int n) . . . 93
4.63.4.7 prepareMessage(SensorValue value, SensorType type) . . . 94
4.64 ev3dev::servo_motor Class Reference . . . 94
4.65 ev3::SignalHandler Class Reference . . . 95
4.66 ev3dev::sound Class Reference . . . 95
4.67 ev3dev::sound_sensor Class Reference . . . 96
4.68 ev3dev::touch_sensor Class Reference . . . 96
4.69 ev3dev::ultrasonic_sensor Class Reference . . . 97

5 File Documentation **99**

5.1 /home/panda/Dokumenty/Repos/Ev3Dev/include/action/Action.h File Reference 99

5.1.1 Detailed Description 100

5.1.2 Typedef Documentation 100

5.1.2.1 StoredActions 100

5.2 /home/panda/Dokumenty/Repos/Ev3Dev/include/action/Behaviour.h File Reference 100

5.2.1 Detailed Description 101

5.3 /home/panda/Dokumenty/Repos/Ev3Dev/include/action/BehaviourState.h File Reference 101

5.3.1 Detailed Description 101

5.4 /home/panda/Dokumenty/Repos/Ev3Dev/include/communication/Communication.h File Reference 101

5.4.1 Detailed Description 102

5.5 /home/panda/Dokumenty/Repos/Ev3Dev/include/communication/CommUtils.h File Reference . . . 102

5.5.1 Detailed Description 102

5.6 /home/panda/Dokumenty/Repos/Ev3Dev/include/communication/Message.h File Reference 102

5.6.1 Detailed Description 103

5.7 /home/panda/Dokumenty/Repos/Ev3Dev/include/robot/Devices.h File Reference 103

5.7.1 Detailed Description 103

Index **105**

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

- ev3::Action 9
  - ev3::ActionDriveDistance 13
  - ev3::ActionDriveForever 15
  - ev3::ActionRepeat 17
  - ev3::ActionRotate 18
    - ev3::ActionRotateRandDirection 20
  - ev3::ActionStop 22
- ev3::Agent 24
- ev3::Behaviour 26
  - ev3::BehaviourDriveOnSquare 30
  - ev3::BehaviourExploreRandom 32
- ev3::BehaviourState 33
- ev3dev::button 36
- ev3::CircularBuffer< T > 36
- ev3::CircularBuffer< std::string > 36
- ev3::ColorUtils 37
- ev3::Command 38
  - ev3::CommandMotor 39
    - ev3::CommandMotorReset 41
    - ev3::CommandMotorRunForever 42
    - ev3::CommandMotorSetSpeed 43
    - ev3::CommandMotorSetSpeedRegEnabled 44
    - ev3::CommandMotorSetStopMode 45
    - ev3::CommandMotorStop 46
  - ev3::CommandSensor 47
- ev3::Communication 48
- ev3::CommUtils 50
- ev3dev::device 53
  - ev3dev::dc_motor 52
  - ev3dev::led 66
  - ev3dev::lego_port 71
  - ev3dev::motor 81
    - ev3dev::large_motor 65
    - ev3dev::medium_motor 74

- ev3dev::power_supply . . . 83
- ev3dev::sensor . . . 90
  - ev3dev::color_sensor . . . 37
  - ev3dev::gyro_sensor . . . 63
  - ev3dev::i2c_sensor . . . 64
  - ev3dev::infrared_sensor . . . 65
  - ev3dev::light_sensor . . . 71
  - ev3dev::sound_sensor . . . 96
  - ev3dev::touch_sensor . . . 96
  - ev3dev::ultrasonic_sensor . . . 97
- ev3dev::servo_motor . . . 94

ev3::Devices . . . 54
ev3::Event . . . 58
- ev3::EventAction . . . 60
- ev3::EventSensorWatch . . . 62

ev3::EventQueue . . . 61
ev3dev::lcd . . . 66
ev3::LedControl . . . 68
ev3::Logger . . . 72
ev3::Master . . . 73
ev3::Message . . . 75
ev3::Motor . . . 80
ev3::CommUtils::NetworkNode . . . 83
ev3::Queue< T > . . . 84
ev3::Queue< ev3::Message > . . . 84
ev3dev::remote_control . . . 84
ev3::Robot . . . 85
- ev3::RobotModelA . . . 86

ev3::RobotState . . . 86
- ev3::RobotStateActive . . . 87
- ev3::RobotStateIdle . . . 88
- ev3::RobotStatePanic . . . 88
- ev3::RobotStatePaused . . . 89
- ev3::RobotStateWorking . . . 89

ev3::Sensor . . . 92
ev3::SignalHandler . . . 95
ev3dev::sound . . . 95

Chapter 2

Class Index

2.1 Class List

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

ev3::Action
Base class for all Action controlling classes 9
ev3::ActionDriveDistance
Implements Robot simple task to drive straight for a given distance 13
ev3::ActionDriveForever
Implements Robot simple task to drive straight forever 15
ev3::ActionRepeat
Stores many Actions in a vector and executes them in loop 17
ev3::ActionRotate
Implements Robot simple task to rotate a given angle, while not driving 18
ev3::ActionRotateRandDirection
Implements Robot simple task to rotate a random angle 20
ev3::ActionStop
Implements Robot simple task to stop all active motors 22
ev3::Agent
Master-side representative of a robot unit 24
ev3::Behaviour
Base class for all defined behaviours 26
ev3::BehaviourDriveOnSquare
Implements complex behaviour of driving on a square-shaped route 30
ev3::BehaviourExploreRandom
Implements complex behaviour of exploring the surrounding with random rotation 32
ev3::BehaviourState
Encapsulates action and other information in a form of a state 33
ev3dev::button 36
ev3::CircularBuffer$<$ T $>$ 36
ev3dev::color_sensor 37
ev3::ColorUtils 37
ev3::Command
Base class for all command controlling classes 38
ev3::CommandMotor
Base class for all motor controlling commands 39
ev3::CommandMotorReset
Calls `reset()` method of containing Motor 41
ev3::CommandMotorRunForever
Calls `run_forever()` method of containing Motor 42

ev3::CommandMotorSetSpeed
Call `set_speed_sp()` method of containing Motor 43
ev3::CommandMotorSetSpeedRegEnabled
Calls `set_speed_regulation_enabled()` method of containing Motor 44
ev3::CommandMotorSetStopMode
Calls `set_stop_command()` method of containing Motor 45
ev3::CommandMotorStop
Calls `stop()` method of containing Motor 46
ev3::CommandSensor
Base class for all sensor controlling commands 47
ev3::Communication
Encapsulates low-level communication and adds logic concerning sending and receiving Message queueing 48
ev3::CommUtils
Responsible for low-level communication 50
ev3dev::dc_motor 52
ev3dev::device 53
ev3::Devices
Singleton class responsible for managing devices connected to the robot 54
ev3::Event
Base class for all Event classes 58
ev3::EventAction
Event class triggered when something happened with Action 60
ev3::EventQueue 61
ev3::EventSensorWatch
Triggered when measurement of certain Sensor occured 62
ev3dev::gyro_sensor 63
ev3dev::i2c_sensor 64
ev3dev::infrared_sensor 65
ev3dev::large_motor 65
ev3dev::lcd 66
ev3dev::led 66
ev3::LedControl
Class specifically designed to eliminate ev3dev library limitations of controlling LED panel . . . 68
ev3dev::lego_port 71
ev3dev::light_sensor 71
ev3::Logger 72
ev3::Master
Controls the whole system and knows about every Agent 73
ev3dev::medium_motor 74
ev3::Message
Stores information passed between physical system units (another robots or master) 75
ev3::Motor
Encapsulates ev3dev::motor 80
ev3dev::motor 81
ev3::CommUtils::NetworkNode
Stores information about a particular node in the network 83
ev3dev::power_supply 83
ev3::Queue< T > 84
ev3dev::remote_control 84
ev3::Robot 85
ev3::RobotModelA 86
ev3::RobotState 86
ev3::RobotStateActive 87
ev3::RobotStateIdle 88
ev3::RobotStatePanic 88
ev3::RobotStatePaused 89
ev3::RobotStateWorking 89

ev3dev::sensor 90
ev3::Sensor
Encapsulates ev3dev::sensor 92
ev3dev::servo_motor 94
ev3::SignalHandler 95
ev3dev::sound 95
ev3dev::sound_sensor 96
ev3dev::touch_sensor 96
ev3dev::ultrasonic_sensor 97

Chapter 3

File Index

3.1 File List

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

/home/panda/Dokumenty/Repos/Ev3Dev/include/action/Action.h
Contains all Action classes 99
/home/panda/Dokumenty/Repos/Ev3Dev/include/action/Behaviour.h
Contains all Behaviour classes 100
/home/panda/Dokumenty/Repos/Ev3Dev/include/action/BehaviourState.h
Contains BehaviourState class 101
/home/panda/Dokumenty/Repos/Ev3Dev/include/action/**Command.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/action/**CommandMotor.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/action/**CommandSensor.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/communication/Communication.h
Contains Communication class 101
/home/panda/Dokumenty/Repos/Ev3Dev/include/communication/CommUtils.h
Contains CommUtils class 102
/home/panda/Dokumenty/Repos/Ev3Dev/include/communication/**Event.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/communication/Message.h
Contains Message class 102
/home/panda/Dokumenty/Repos/Ev3Dev/include/control/**LedControl.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/**ev3dev.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/master/**Agent.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/master/**Master.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/robot/Devices.h
Contains Devices classes 103
/home/panda/Dokumenty/Repos/Ev3Dev/include/robot/**Motor.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/robot/**Robot.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/robot/**RobotModelA.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/robot/**RobotState.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/robot/**Sensor.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/utils/**CircularBuffer.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/utils/**ColorUtils.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/utils/**EventQueue.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/utils/**Logger.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/utils/**Queue.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/utils/**SignalHandler.h** . . . . ??
/home/panda/Dokumenty/Repos/Ev3Dev/include/utils/**Utils.h** . . . . ??

Chapter 4

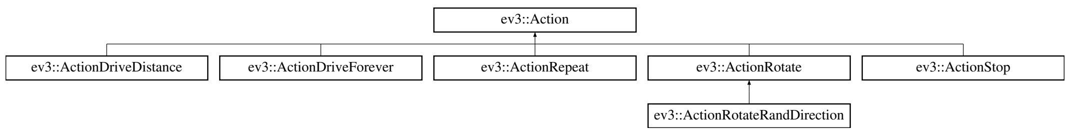
Class Documentation

4.1 ev3::Action Class Reference

Base class for all Action controlling classes.

```
#include <Action.h>
```

Inheritance diagram for ev3::Action:



Public Types

- enum ActionType {
  NOP, REPEAT, DRIVE_DISTANCE, ROTATE,
  ROTATE_RANDOM_DIR, STOP, DRIVE_FOREVER }

  Type of Action.
- typedef std::function$<$ bool(void) $>$ EndCondition

  Type for lambda functions to store end of Action condition.

Public Member Functions

- Action (CommandsVector commands, ActionType type)

  Constructor with CommandsVector and ActionType parameters.
- Action (CommandsVector commands)

  Constructor with CommandsVector parameter.
- Action (ActionType type)

  Constructor with ActionType parameter.
- virtual ∼Action ()

  Default destructor.
- virtual void execute ()

  Executes stored Commands in a sequence.

- virtual bool isFinished ()

    Check if Action condition is fullfilled.
- virtual bool isExecuted ()

    Check if action was executed.
- virtual std::string getActionPrototype ()

    Generate std::string prototype for Action.
- virtual std::string getString ()

    Get human-readable Action name.
- void setCommands (CommandsVector commands)

    Set Commands to be executed.
- void setEndCondition (EndCondition condition)

    Set end condition for Action.
- ActionType getType ()

    Get current Action type.

Static Public Attributes

- static const std::string EMPTY_PROTO

    String for empty Action prototype.

Protected Attributes

- ActionType _type

    Action type.
- CommandsVector _commands

    Vector of Commands.
- EndCondition _endCondition

    Lambda function defining Action end condition.
- bool _isExecuted = false

    True if action is already executed, false otherwise.

4.1.1 Detailed Description

Base class for all Action controlling classes.

Each Action contains of a sequence of many Commands and all of them are executed immediately, one after another. Action is valid, until specific Event occurs or its endCondition function returns true.

Action objects are instantiated accordingly to Robot model that uses them. Actions are predefined and cannot be dynamically created.

4.1.2 Member Enumeration Documentation

4.1.2.1 enum **ev3::Action::ActionType**

Type of Action.

It directly points to derived class being used.

See also

Robot::AvailableActions

Enumerator

NOP No operation.

REPEAT Repeats execution of other Actions.

DRIVE_DISTANCE Power Motor to reach certain distance.

ROTATE Rotate Robot for given angle.

ROTATE_RANDOM_DIR Rotate for given angle, clockwise or counterclockwise at random.

STOP Stop all active motors.

DRIVE_FOREVER Drive forward or backward infinetely.

4.1.3 Constructor & Destructor Documentation

4.1.3.1 Action::Action (**CommandsVector** ***commands***, **ActionType** ***type***)

Constructor with CommandsVector and ActionType parameters.

Parameters

commands	Commands stored within this Action.
type	Type of Action used.

4.1.3.2 Action::Action (**CommandsVector** ***commands***)

Constructor with CommandsVector parameter.

Action type is set to Action::NOP .

Parameters

commands	Commands stored within this Action.

4.1.3.3 Action::Action (**ActionType** ***type***)

Constructor with ActionType parameter.

Parameters

type	Type of Action used.

4.1.4 Member Function Documentation

4.1.4.1 std::string Action::getActionPrototype ( ) [virtual]

Generate std::string prototype for Action.

Returns

Encoded Action data into std::string.

Reimplemented in ev3::ActionDriveForever, ev3::ActionStop, ev3::ActionRotateRandDirection, ev3::ActionRotate, and ev3::ActionDriveDistance.

4.1.4.2 std::string Action::getString ( ) [virtual]

Get human-readable Action name.

Returns

String containing Action name.

Reimplemented in ev3::ActionDriveForever, ev3::ActionStop, ev3::ActionRotateRandDirection, ev3::ActionRotate, ev3::ActionDriveDistance, and ev3::ActionRepeat.

4.1.4.3 Action::ActionType Action::getType ( )

Get current Action type.

Returns

ActionType value.

4.1.4.4 bool Action::isExecuted ( ) [virtual]

Check if action was executed.

Returns

True if actcion was already executed, false otherwise.

4.1.4.5 bool Action::isFinished ( ) [virtual]

Check if Action condition is fullfilled.

Returns

Value returned from Action::_endCondition.

4.1.4.6 void Action::setCommands (CommandsVector *commands*)

Set Commands to be executed.

Parameters

commands	CommandsVector with pointers to commands.

4.1.4.7 void Action::setEndCondition (EndCondition *condition*)

Set end condition for Action.

Parameters

condition	Lambda function returning bool value.

4.1.5 Member Data Documentation

4.1.5.1 EndCondition ev3::Action::_endCondition [protected]

Initial value:

```
= [] ()
        {
            return true;
        }
```

Lambda function defining Action end condition.

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

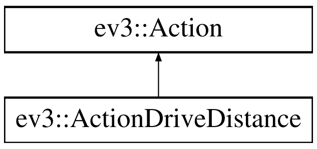
- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/Action.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/Action.cpp

4.2 ev3::ActionDriveDistance Class Reference

Implements Robot simple task to drive straight for a given distance.

`#include <Action.h>`

Inheritance diagram for ev3::ActionDriveDistance:



Public Member Functions

- ActionDriveDistance (int distance)
  Constructor with distance parameter.
- ActionDriveDistance (CommandsVector commands, int distance)
  Constructor with CommandsVector and distance parameters.
- int getDistance ()
  Get distance the Robot has to drive.
- virtual std::string getActionPrototype ()
  Get ActionDriveDistance encoded name and its parameters.
- virtual std::string getString () override
  Get ActionDriveDistance human-readable name.

Additional Inherited Members

4.2.1 Detailed Description

Implements Robot simple task to drive straight for a given distance.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 ActionDriveDistance::ActionDriveDistance (int *distance*)

Constructor with distance parameter.

Parameters

distance	Integer value in Robot units to be driven.

4.2.2.2 ActionDriveDistance::ActionDriveDistance (CommandsVector *commands,* int *distance*)

Constructor with CommandsVector and distance parameters.

Parameters

commands	Sequence of commands to be executed.
distance	Integer value in Robot units to be driven.

4.2.3 Member Function Documentation

4.2.3.1 std::string ActionDriveDistance::getActionPrototype ( ) [virtual]

Get ActionDriveDistance encoded name and its parameters.

Returns

String with encoded name and parameters.

Reimplemented from ev3::Action.

4.2.3.2 int ActionDriveDistance::getDistance ( )

Get distance the Robot has to drive.

Returns

Integer value in Robot units.

4.2.3.3 std::string ActionDriveDistance::getString ( ) `[override], [virtual]`

Get ActionDriveDistance human-readable name.

Returns

String with name and parameters

Reimplemented from ev3::Action.

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

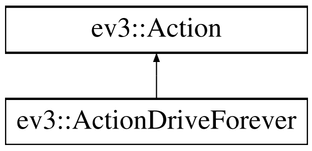
- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/Action.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/Action.cpp

4.3 ev3::ActionDriveForever Class Reference

Implements Robot simple task to drive straight forever.

```
#include <Action.h>
```

Inheritance diagram for ev3::ActionDriveForever:



Public Member Functions

- ActionDriveForever (bool forward=true)

  Constructor with direction parameter.
- ActionDriveForever (CommandsVector commands, bool forward=true)

  Constructor with CommandsVector and direction parameter.
- virtual std::string getActionPrototype ()

  Get ActionDriveForever encoded name and its parameters.
- virtual std::string getString () override

  Get ActionDriveForever human-readable name.
- bool isForward ()

  Return specified direction.

Additional Inherited Members

4.3.1 Detailed Description

Implements Robot simple task to drive straight forever.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 ActionDriveForever::ActionDriveForever (bool *forward* = `true`)

Constructor with direction parameter.

Parameters

forward	True to drive forward, false otherwise.

4.3.2.2 ActionDriveForever::ActionDriveForever (CommandsVector *commands*, bool *forward* = `true`)

Constructor with CommandsVector and direction parameter.

Parameters

commands	Sequence of commands to be executed.
forward	True to drive forward, false otherwise.

4.3.3 Member Function Documentation

4.3.3.1 std::string ActionDriveForever::getActionPrototype ( ) `[virtual]`

Get ActionDriveForever encoded name and its parameters.

Returns

String with encoded name and parameters.

Reimplemented from ev3::Action.

4.3.3.2 std::string ActionDriveForever::getString ( ) [override], [virtual]

Get ActionDriveForever human-readable name.

Returns

String with name and parameters

Reimplemented from ev3::Action.

4.3.3.3 bool ActionDriveForever::isForward ( )

Return specified direction.

Returns

True for forward, false for backward.

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

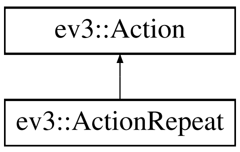
- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/Action.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/Action.cpp

4.4 ev3::ActionRepeat Class Reference

Stores many Actions in a vector and executes them in loop.

`#include <Action.h>`

Inheritance diagram for ev3::ActionRepeat:



Public Member Functions

- ActionRepeat (StoredActions actions, unsigned int n)
  Constructor with StoredActions and iterations parameters.
- virtual void execute ()
  Continue with executing stored Actions.
- virtual std::string getString ()
  Return human-readable ActionRepeat name.

Additional Inherited Members

4.4.1 Detailed Description

Stores many Actions in a vector and executes them in loop.

Number of iterations is given and may be infinite.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 ActionRepeat::ActionRepeat (StoredActions *actions*, unsigned int *n*)

Constructor with StoredActions and iterations parameters.

Parameters

actions	Vector of Actions to be executed in a loop.
n	Number of iterations. If 0 is given, loop will be infinite.

4.4.3 Member Function Documentation

4.4.3.1 std::string ActionRepeat::getString ( ) `[virtual]`

Return human-readable ActionRepeat name.

Returns

String containing Action name.

Reimplemented from ev3::Action.

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

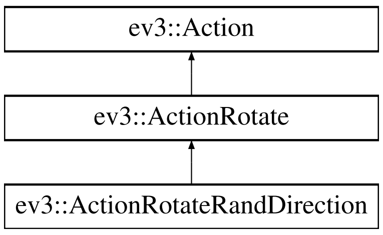
- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/Action.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/Action.cpp

4.5 ev3::ActionRotate Class Reference

Implements Robot simple task to rotate a given angle, while not driving.

`#include <Action.h>`

Inheritance diagram for ev3::ActionRotate:



Public Member Functions

- ActionRotate (int rotation)

  Constructor with rotation parameter in degrees.
- ActionRotate (CommandsVector commands, int rotation)

  Constructor with CommandsVector and rotation parameters.
- int getRotation ()

  Get Robot rotation.
- virtual std::string getActionPrototype ()

  Get ActionRotate encoded name and its parameters.
- virtual std::string getString () override

  Get ActionRotate human-readable name.

Protected Attributes

- int _rotation

  Angle of rotation in degrees for the Robot.

Additional Inherited Members

4.5.1 Detailed Description

Implements Robot simple task to rotate a given angle, while not driving.

Rotation is made in place.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 ActionRotate::ActionRotate (int *rotation*)

Constructor with rotation parameter in degrees.

Parameters

rotation	Number of degrees to rotate. Positive value rotates clockwise.

4.5.2.2 ActionRotate::ActionRotate (CommandsVector *commands,* int *rotation*)

Constructor with CommandsVector and rotation parameters.

Parameters

commands	Sequence of commands to be executed.
rotation	Integer value of Robot rotation in degrees.

4.5.3 Member Function Documentation

4.5.3.1 std::string ActionRotate::getActionPrototype ( ) [virtual]

Get ActionRotate encoded name and its parameters.

Returns

String with encoded name and parameters.

Reimplemented from ev3::Action.

Reimplemented in ev3::ActionRotateRandDirection.

4.5.3.2 int ActionRotate::getRotation ( )

Get Robot rotation.

Returns

Integer value of rotation in degrees.

4.5.3.3 std::string ActionRotate::getString ( ) [override], [virtual]

Get ActionRotate human-readable name.

Returns

String with name and parameters

Reimplemented from ev3::Action.

Reimplemented in ev3::ActionRotateRandDirection.

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

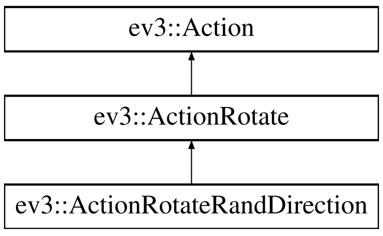
- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/Action.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/Action.cpp

4.6 ev3::ActionRotateRandDirection Class Reference

Implements Robot simple task to rotate a random angle.

`#include <Action.h>`

Inheritance diagram for ev3::ActionRotateRandDirection:



Public Member Functions

- ActionRotateRandDirection (int rotation)

    Constructor with rotation parameter in degrees.
- ActionRotateRandDirection (CommandsVector commands, int rotation)

    Constructor with CommandsVector and rotation parameters.
- virtual std::string getActionPrototype ()

    Get ActionRotateRandDirection encoded name and its parameters.
- virtual std::string getString () override

    Get ActionRotateRandDirection human-readable name.
- virtual void execute () override

Additional Inherited Members

4.6.1 Detailed Description

Implements Robot simple task to rotate a random angle.

Rotation is performed in place. Maximum angle in degrees is passed via constructor argument.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 ActionRotateRandDirection::ActionRotateRandDirection (int *rotation*)

Constructor with rotation parameter in degrees.

Parameters

rotation	Upper limit of degrees to rotate randomly. Positive value rotates clockwise.

4.6.2.2 ActionRotateRandDirection::ActionRotateRandDirection (CommandsVector *commands*, int *rotation*)

Constructor with CommandsVector and rotation parameters.

Parameters

commands	Sequence of commands to be executed.
rotation	Upper limit of degrees to rotate randomly. Positive value rotates clockwise.

4.6.3 Member Function Documentation

4.6.3.1 void ActionRotateRandDirection::execute ( ) `[override]`, `[virtual]`

See also

Action::execute

Reimplemented from ev3::Action.

4.6.3.2 std::string ActionRotateRandDirection::getActionPrototype ( ) [virtual]

Get ActionRotateRandDirection encoded name and its parameters.

Returns

String with encoded name and parameters.

Reimplemented from ev3::ActionRotate.

4.6.3.3 std::string ActionRotateRandDirection::getString ( ) [override], [virtual]

Get ActionRotateRandDirection human-readable name.

Returns

String with name and parameters

Reimplemented from ev3::ActionRotate.

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

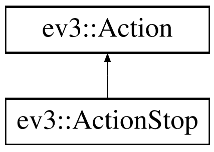
- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/Action.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/Action.cpp

4.7 ev3::ActionStop Class Reference

Implements Robot simple task to stop all active motors.

```
#include <Action.h>
```

Inheritance diagram for ev3::ActionStop:



Public Member Functions

- ActionStop ()

    Default constructor.
- ActionStop (CommandsVector commands)

    Constructor with CommandsVector parameter.
- virtual std::string getActionPrototype ()

    Get ActionStop encoded name and its parameters.
- virtual std::string getString () override

    Get ActionStop human-readable name.

Additional Inherited Members

4.7.1 Detailed Description

Implements Robot simple task to stop all active motors.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 ActionStop::ActionStop (CommandsVector *commands*)

Constructor with CommandsVector parameter.

Parameters

commands	Sequence of commands to be executed.

4.7.3 Member Function Documentation

4.7.3.1 std::string ActionStop::getActionPrototype ( ) `[virtual]`

Get ActionStop encoded name and its parameters.

Returns

String with encoded name and parameters.

Reimplemented from ev3::Action.

4.7.3.2 std::string ActionStop::getString ( ) `[override]`, `[virtual]`

Get ActionStop human-readable name.

Returns

String with name and parameters

Reimplemented from ev3::Action.

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/Action.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/Action.cpp

4.8 ev3::Agent Class Reference

Master-side representative of a robot unit.

```
#include <Agent.h>
```

Public Member Functions

- unsigned int getId ()

  Agent id getter.
- void setId (const unsigned int id)

  Agent id setter.
- unsigned int getCommId ()

  Current communication id getter.
- void setCommId (const unsigned int commId)

  Communication id setter.
- void processMessage (Message ∗message, Message ∗retMessage)

  Process received Message to produce response.
- void updateLastMessage (Message ∗message)

  Update data concerning last message sent.
- void setBehaviour (SharedPtrBehaviour behaviour)

  Set currently executing Behaviour.
- void setMeasurement (Measurements measurements)

  Set measurements that must be done on corresponding Robot.

4.8.1 Detailed Description

Master-side representative of a robot unit.

Lacks all device references and action execution.

4.8.2 Member Function Documentation

4.8.2.1 unsigned int Agent::getCommId ( )

Current communication id getter.

Returns

Id of Message id synchronised between Agent and Robot.

4.8.2.2 unsigned int Agent::getId ( )

Agent id getter.

Returns

Id given by Master.

4.8.2.3 void Agent::processMessage (**Message** ∗ *message,* **Message** ∗ *retMessage*)

Process received Message to produce response.

Parameters

message	Message to be analyzed.
retMessage	Modified Message to be sent to Robot.

4.8.2.4 void Agent::setBehaviour (**SharedPtrBehaviour** *behaviour*)

Set currently executing Behaviour.

Parameters

behaviour	Behaviour shared_ptr object.

4.8.2.5 void Agent::setCommId (const unsigned int *commId*)

Communication id setter.

Parameters

comm↩ Id	New communication id.

4.8.2.6 void Agent::setId (const unsigned int *id*)

Agent id setter.

Parameters

id	New id for this Agent.

4.8.2.7 void Agent::setMeasurement (Measurements *measurements*)

Set measurements that must be done on corresponding Robot.

Parameters

measurements	Vector of Sensor types.

4.8.2.8 void Agent::updateLastMessage (Message ∗ *message*)

Update data concerning last message sent.

Parameters

message	Last Message sent to corresponding Robot.

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

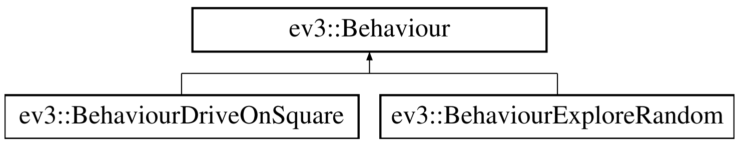
- /home/panda/Dokumenty/Repos/Ev3Dev/include/master/Agent.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/master/Agent.cpp

4.9 ev3::Behaviour Class Reference

Base class for all defined behaviours.

```
#include <Behaviour.h>
```

Inheritance diagram for ev3::Behaviour:



Public Types

- enum BehaviourType { CUSTOM, DRIVE_ON_SQUARE, EXPLORE_RANDOM }

  Type of Behaviour.

Public Member Functions

- Behaviour ()=default

  Default constructor.
- Behaviour (BehaviourType type, BehaviourStates states)

  Constructor with type and states vector parameters.
- Behaviour (BehaviourType type)

  Constructor with behaviour type.
- void setStates (BehaviourStates states)

  Available states setter.
- void setReactionStates (BehaviourStates reactionStates)

  Special reaction states which occur when event is fired.
- void setStopState (BehaviourState state)

  Special stop state, used mainly to get precise sensor measurements.
- void setMeasurements (Measurements measurements)

  Sensor which measurements will be required.
- virtual StringVector getPrototype ()

  Get Behaviour encoded name and its parameters.
- virtual std::string getString ()

  Get Behaviour human-readable name.
- virtual void process ()

  Updates behaviour in every iteration.
- void stop ()

  Stops Behaviour execution definetely.
- void pause ()

  Pauses Behaviour execution until it's resumed.
- void resume ()

  Resumes paused Behaviour.
- void start ()

  Starts Behaviour execution.
- void react (Event::EventType type)

  Performs special actions based on Event passed.

Protected Attributes

- BehaviourType _type

  Type of Behaviour.
- BehaviourState _currentState

  Currently processed Behaviour.
- BehaviourState _stopState

  Special stop state for measurements and accurate action execution.
- BehaviourStates _states

  Vector with all Behaviour available states.
- BehaviourStates _reactionStates

  Vector with all reaction states, occuring after specific events.
- Measurements _measurements

  Vector of all Sensor ids that will be measured.
- bool _active = false

  Specified whether Behaviour is currently active or not.

4.9.1 Detailed Description

Base class for all defined behaviours.

It's responsible for maintaining active actions in a form of a state machine as well as keep track of sensors' measurements.

4.9.2 Member Enumeration Documentation

4.9.2.1 enum ev3::Behaviour::BehaviourType

Type of Behaviour.

Enumerator

CUSTOM	Custom, user-defined behaviour.
DRIVE_ON_SQUARE	Follow square-shaped route.
EXPLORE_RANDOM	Drive in a direction and rotate randomly.

4.9.3 Constructor & Destructor Documentation

4.9.3.1 Behaviour::Behaviour (**BehaviourType** *type,* **BehaviourStates** *states*)

Constructor with type and states vector parameters.

Parameters

type	Behaviour type.
states	Vector of available Behaviour states.

4.9.3.2 Behaviour::Behaviour (**BehaviourType** *type*)

Constructor with behaviour type.

Parameters

type	Behaviour type.

4.9.4 Member Function Documentation

4.9.4.1 StringVector Behaviour::getPrototype ( ) `[virtual]`

Get Behaviour encoded name and its parameters.

Returns

StringVector with encoded name and parameters as its members.

Reimplemented in ev3::BehaviourExploreRandom, and ev3::BehaviourDriveOnSquare.

4.9.4.2 std::string Behaviour::getString ( ) [virtual]

Get Behaviour human-readable name.

Returns

String with name and parameters

Reimplemented in ev3::BehaviourExploreRandom, and ev3::BehaviourDriveOnSquare.

4.9.4.3 void Behaviour::react (Event::EventType *type*)

Performs special actions based on Event passed.

Parameters

type	Event type that will be processed.

4.9.4.4 void Behaviour::setMeasurements (Measurements *measurements*)

Sensor which measurements will be required.

Parameters

measurements	Vector of sensor types.

4.9.4.5 void Behaviour::setReactionStates (BehaviourStates *reactionStates*)

Special reaction states which occur when event is fired.

Parameters

reactionStates	Vector of reaction states for this Behaviour.

4.9.4.6 void Behaviour::setStates (BehaviourStates *states*)

Available states setter.

Parameters

states	Vector of states for this Behaviour.

4.9.4.7 void Behaviour::setStopState (BehaviourState *state*)

Special stop state, used mainly to get precise sensor measurements.

Parameters

state	BehaviourState object for stop state.

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

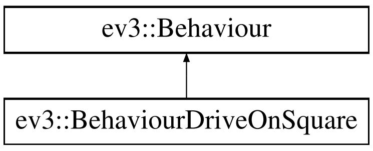
- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/Behaviour.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/Behaviour.cpp

4.10 ev3::BehaviourDriveOnSquare Class Reference

Implements complex behaviour of driving on a square-shaped route.

```
#include <Behaviour.h>
```

Inheritance diagram for ev3::BehaviourDriveOnSquare:



Public Member Functions

- BehaviourDriveOnSquare (unsigned int side, bool turningRight)

    Constructor with square side and direction (either left or right).
- BehaviourDriveOnSquare (BehaviourStates states, unsigned int side, bool turningRight)

    Constructor with Behaviour states and driving parameters.
- virtual StringVector getPrototype ()

    Get BehaviourDriveOnSquare encoded name and its parameters.
- virtual std::string getString ()

    Get BehaviourDriveOnSquare human-readable name.

Additional Inherited Members

4.10.1 Detailed Description

Implements complex behaviour of driving on a square-shaped route.

Square side and direction (right/left) can be implicitly defined.

4.10.2 Constructor & Destructor Documentation

4.10.2.1 BehaviourDriveOnSquare::BehaviourDriveOnSquare (unsigned int *side*, bool *turningRight*)

Constructor with square side and direction (either left or right).

Parameters

side	Length of square side in units.
turningRight	True for turning right, false otherwise.

4.10.2.2 BehaviourDriveOnSquare::BehaviourDriveOnSquare (BehaviourStates *states*, unsigned int *side*, bool *turningRight*)

Constructor with Behaviour states and driving parameters.

Parameters

states	Vector of Behaviour states to be processed.
side	Length of square side in units.
turningRight	True for turning right, false otherwise.

4.10.3 Member Function Documentation

4.10.3.1 StringVector BehaviourDriveOnSquare::getPrototype ( ) `[virtual]`

Get BehaviourDriveOnSquare encoded name and its parameters.

Returns

StringVector with encoded name and parameters as its members.

Reimplemented from ev3::Behaviour.

4.10.3.2 std::string BehaviourDriveOnSquare::getString ( ) `[virtual]`

Get BehaviourDriveOnSquare human-readable name.

Returns

String with name and parameters

Reimplemented from ev3::Behaviour.

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

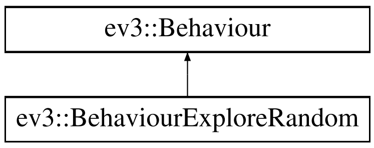
- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/Behaviour.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/Behaviour.cpp

4.11 ev3::BehaviourExploreRandom Class Reference

Implements complex behaviour of exploring the surrounding with random rotation.

```
#include <Behaviour.h>
```

Inheritance diagram for ev3::BehaviourExploreRandom:



Public Member Functions

- BehaviourExploreRandom ()

  Default constructor.
- BehaviourExploreRandom (BehaviourStates states)

  Constructor with Behaviour states parameter.
- virtual StringVector getPrototype ()

  Get BehaviourExploreRandom encoded name and its parameters.
- virtual std::string getString ()

  Get BehaviourExploreRandom human-readable name.

Additional Inherited Members

4.11.1 Detailed Description

Implements complex behaviour of exploring the surrounding with random rotation.

4.11.2 Constructor & Destructor Documentation

4.11.2.1 BehaviourExploreRandom::BehaviourExploreRandom (BehaviourStates *states*)

Constructor with Behaviour states parameter.

Parameters

states	Vector of available Behaviour states.

4.11.3 Member Function Documentation

4.11.3.1 StringVector BehaviourExploreRandom::getPrototype ( ) [virtual]

Get BehaviourExploreRandom encoded name and its parameters.

Returns

StringVector with encoded name and parameters as its members.

Reimplemented from ev3::Behaviour.

4.11.3.2 std::string BehaviourExploreRandom::getString ( ) `[virtual]`

Get BehaviourExploreRandom human-readable name.

Returns

String with name and parameters

Reimplemented from ev3::Behaviour.

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/Behaviour.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/Behaviour.cpp

4.12 ev3::BehaviourState Class Reference

Encapsulates action and other information in a form of a state.

```
#include <BehaviourState.h>
```

Public Member Functions

- BehaviourState ()=default

  Default constructor.
- BehaviourState (const BehaviourState &)=default

  Default copy constructor.
- BehaviourState (SharedPtrAction action, unsigned int nextState, bool isStopState=false)

  Constructor with action, next state id and stop state flag.
- BehaviourState (SharedPtrAction action, unsigned int nextState, ReactionsTransitions reactions)

  Constructor with action, next state id and event-state map.
- unsigned int process ()

  Process state in every iteration.
- SharedPtrAction getAction ()

  State's Action getter.
- void setNextState (const unsigned int next)

  Next state id setter.
- bool isStopState ()

  Stop flag getter.
- void setReactions (ReactionsTransitions reactions)

  Reactions setter.
- int getReaction (Event::EventType type)

  Reaction getter.

4.12.1 Detailed Description

Encapsulates action and other information in a form of a state.

It can contain reactions to different events.

4.12.2 Constructor & Destructor Documentation

4.12.2.1 ev3::BehaviourState::BehaviourState (const **BehaviourState** &) [default]

Default copy constructor.

Parameters

Other	BehaviourState object.

4.12.2.2 BehaviourState::BehaviourState (**SharedPtrAction** *action*, unsigned int *nextState*, bool *isStopState* = false)

Constructor with action, next state id and stop state flag.

Parameters

action	Action object to be executed within this state.
nextState	Id of the next state that will replace this one.
isStopState	Flag defining this state as a in-between, stopping state.

4.12.2.3 BehaviourState::BehaviourState (**SharedPtrAction** *action*, unsigned int *nextState*, **ReactionsTransitions** *reactions*)

Constructor with action, next state id and event-state map.

Parameters

action	Action object to be executed within this state,
nextState	Id of the next state that will replace this one.
reactions	Map containing event-state pairs describing reactions.

4.12.3 Member Function Documentation

4.12.3.1 **SharedPtrAction** BehaviourState::getAction ( )

State's Action getter.

Returns

Action shared_ptr object.

4.12.3.2 int BehaviourState::getReaction (Event::EventType *type*)

Reaction getter.

Parameters

type	EventType to which reaction occurs.

Returns

Id of the reaction state.

4.12.3.3 bool BehaviourState::isStopState ( )

Stop flag getter.

Returns

True if state is flagged as a stop state, false otherwise.

4.12.3.4 unsigned int BehaviourState::process ( )

Process state in every iteration.

Returns

Id of the next state.

4.12.3.5 void BehaviourState::setNextState (const unsigned int *next*)

Next state id setter.

Parameters

next	Integer defining next state id.

4.12.3.6 void BehaviourState::setReactions (ReactionsTransitions *reactions*)

Reactions setter.

Parameters

reactions	Map with Event-State pair.

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/BehaviourState.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/BehaviourState.cpp

4.13 ev3dev::button Class Reference

Public Member Functions

- **button** (int bit)
- bool **pressed** () const
- bool **process** ()

Static Public Member Functions

- static bool **process_all** ()

Public Attributes

- std::function< void(bool)> **onclick**

Static Public Attributes

- static button **back**
- static button **left**
- static button **right**
- static button **up**
- static button **down**
- static button **enter**

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.14 ev3::CircularBuffer< T > Class Template Reference

Public Member Functions

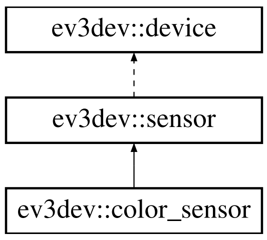
- **CircularBuffer** (unsigned int limit)
- void **push** (T object)
- bool **contain** (T object)

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/utils/CircularBuffer.h

4.15 ev3dev::color_sensor Class Reference

Inheritance diagram for ev3dev::color_sensor:



Public Member Functions

- **color_sensor** (address_type address=INPUT_AUTO)
- int **reflected_light_intensity** ()
- int **ambient_light_intensity** ()
- int **color** ()
- int **red** ()
- int **green** ()
- int **blue** ()

Static Public Attributes

- static const std::string **mode_col_reflect** { "COL-REFLECT" }
- static const std::string **mode_col_ambient** { "COL-AMBIENT" }
- static const std::string **mode_col_color** { "COL-COLOR" }
- static const std::string **mode_ref_raw** { "REF-RAW" }
- static const std::string **mode_rgb_raw** { "RGB-RAW" }

Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.16 ev3::ColorUtils Class Reference

Public Types

- typedef std::string **colorCode**

Static Public Member Functions

- static void **printColorTest** ()

Static Public Attributes

- static const colorCode **BLACK** {"\033[30m"}
- static const colorCode **RED** {"\033[31m"}
- static const colorCode **GREEN** {"\033[32m"}
- static const colorCode **YELLOW** {"\033[33m"}
- static const colorCode **BLUE** {"\033[34m"}
- static const colorCode **MAGENTA** {"\033[35m"}
- static const colorCode **CYAN** {"\033[36m"}
- static const colorCode **WHITE** {"\033[37m"}
- static const colorCode **BLACK_BOLD** {"\033[30;1m"}
- static const colorCode **RED_BOLD** {"\033[31;1m"}
- static const colorCode **GREEN_BOLD** {"\033[32;1m"}
- static const colorCode **YELLOW_BOLD** {"\033[33;1m"}
- static const colorCode **BLUE_BOLD** {"\033[34;1m"}
- static const colorCode **MAGENTA_BOLD** {"\033[35;1m"}
- static const colorCode **CYAN_BOLD** {"\033[36;1m"}
- static const colorCode **WHITE_BOLD** {"\033[37;1m"}
- static const colorCode **BLACK_FAINT** {"\033[30;2m"}
- static const colorCode **RED_FAINT** {"\033[31;2m"}
- static const colorCode **GREEN_FAINT** {"\033[32;2m"}
- static const colorCode **YELLOW_FAINT** {"\033[33;2m"}
- static const colorCode **BLUE_FAINT** {"\033[34;2m"}
- static const colorCode **MAGENTA_FAINT** {"\033[35;2m"}
- static const colorCode **CYAN_FAINT** {"\033[36;2m"}
- static const colorCode **WHITE_FAINT** {"\033[37;2m"}
- static const colorCode **RESET** {"\033[39;0m"}

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

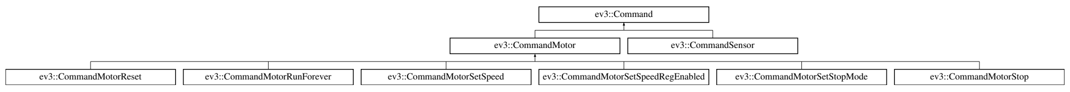
- /home/panda/Dokumenty/Repos/Ev3Dev/include/utils/ColorUtils.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/utils/ColorUtils.cpp

4.17 ev3::Command Class Reference

Base class for all command controlling classes.

`#include <Command.h>`

Inheritance diagram for ev3::Command:


ev3::Command
ev3::CommandMotor
ev3::CommandSensor
ev3::CommandMotorReset
ev3::CommandMotorRunForever
ev3::CommandMotorSetSpeed
ev3::CommandMotorSetSpeedRegEnabled
ev3::CommandMotorSetStopMode
ev3::CommandMotorStop

Public Member Functions

- Command ()

  Default constructor.
- virtual void execute ()

  Execute device specific command.
- virtual std::string getString ()

  Return Command's name.

Protected Attributes

- std::string _debugInfo = ""

    String containing Command's name.

4.17.1 Detailed Description

Base class for all command controlling classes.

Each Command class encapsulates basic motor or sensor operation.

4.17.2 Member Function Documentation

4.17.2.1 std::string Command::getString ( ) [virtual]

Return Command's name.

Returns

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

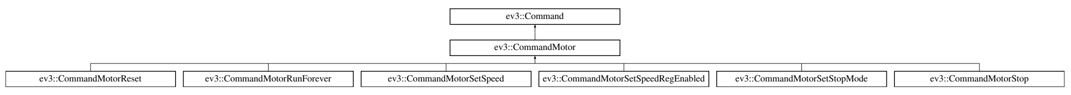
- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/Command.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/Command.cpp

4.18 ev3::CommandMotor Class Reference

Base class for all motor controlling commands.

`#include <CommandMotor.h>`

Inheritance diagram for ev3::CommandMotor:


ev3::Command
ev3::CommandMotor
ev3::CommandMotorReset
ev3::CommandMotorRunForever
ev3::CommandMotorSetSpeed
ev3::CommandMotorSetSpeedRegEnabled
ev3::CommandMotorSetStopMode
ev3::CommandMotorStop

Public Member Functions

- CommandMotor (Motor &motor)

    Constructor with ev3dev::motor parameter.

- Motor getMotor ()

    Get motor associated with Command.

Protected Attributes

- const std::string SPEED_REGULATION_ON = "on"

  Command parameter to turn speed regulation on a Motor on.
- const std::string SPEED_REGULATION_OFF = "off"

  Command parameter to turn speed regulation on a Motor off.
- Motor _motor

  Motor on which this CommandMotor will be executed.

4.18.1 Detailed Description

Base class for all motor controlling commands.

See also

ev3dev::motor

4.18.2 Constructor & Destructor Documentation

4.18.2.1 CommandMotor::CommandMotor (Motor & *motor*)

Constructor with ev3dev::motor parameter.

Parameters

motor	Motor to execute CommandMotor on.

4.18.3 Member Function Documentation

4.18.3.1 Motor CommandMotor::getMotor ( )

Get motor associated with Command.

Returns

Motor class object.

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

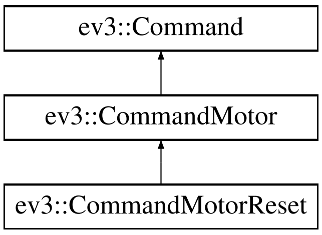
- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/CommandMotor.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/CommandMotor.cpp

4.19 ev3::CommandMotorReset Class Reference

Calls `reset()` method of containing Motor.

`#include <CommandMotor.h>`

Inheritance diagram for ev3::CommandMotorReset:



Public Member Functions

- CommandMotorReset (Motor &motor)

  Constructor with ev3dev::motor parameter.
- void execute () override

  Perform `reset()` method on Motor.

Additional Inherited Members

4.19.1 Detailed Description

Calls `reset()` method of containing Motor.

4.19.2 Constructor & Destructor Documentation

4.19.2.1 CommandMotorReset::CommandMotorReset (Motor & *motor*)

Constructor with ev3dev::motor parameter.

Parameters

motor	Motor to execute CommandMotor on.

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

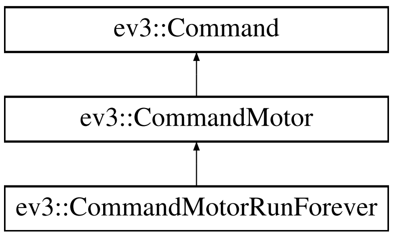
- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/CommandMotor.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/CommandMotor.cpp

4.20 ev3::CommandMotorRunForever Class Reference

Calls `run_forever()` method of containing Motor.

`#include <CommandMotor.h>`

Inheritance diagram for ev3::CommandMotorRunForever:



Public Member Functions

- CommandMotorRunForever (Motor &motor)

  Constructor with ev3dev::motor parameter.
- void execute () override

  Perform `run_forever()` method on Motor.

Additional Inherited Members

4.20.1 Detailed Description

Calls `run_forever()` method of containing Motor.

4.20.2 Constructor & Destructor Documentation

4.20.2.1 CommandMotorRunForever::CommandMotorRunForever (Motor & *motor*)

Constructor with ev3dev::motor parameter.

Parameters

motor	Motor to execute CommandMotor on.

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

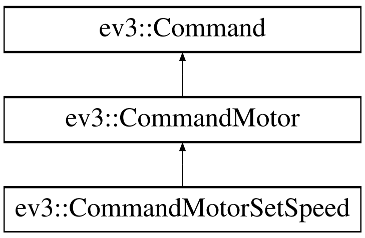
- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/CommandMotor.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/CommandMotor.cpp

4.21 ev3::CommandMotorSetSpeed Class Reference

Call `set_speed_sp()` method of containing Motor.

`#include <CommandMotor.h>`

Inheritance diagram for ev3::CommandMotorSetSpeed:



Public Member Functions

- CommandMotorSetSpeed (Motor &motor, int value)

  Constructor with ev3dev::motor parameter.
- void execute () override

  Perform `set_speed_sp()` method on Motor.

Additional Inherited Members

4.21.1 Detailed Description

Call `set_speed_sp()` method of containing Motor.

4.21.2 Constructor & Destructor Documentation

4.21.2.1 CommandMotorSetSpeed::CommandMotorSetSpeed (Motor & *motor,* int *value*)

Constructor with ev3dev::motor parameter.

Parameters

motor	Motor to execute CommandMotor on.
value	Speed value in tacho pulses per second.

Warning

Speed regulation must be turned on for this to take effect.

See also

CommandMotorSetSpeedRegEnabled

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

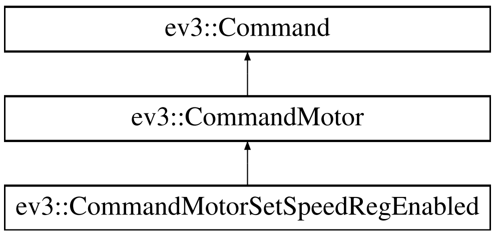
- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/CommandMotor.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/CommandMotor.cpp

4.22 ev3::CommandMotorSetSpeedRegEnabled Class Reference

Calls `set_speed_regulation_enabled()` method of containing Motor.

`#include <CommandMotor.h>`

Inheritance diagram for ev3::CommandMotorSetSpeedRegEnabled:



Public Member Functions

- CommandMotorSetSpeedRegEnabled (Motor &motor, bool value)

    Constructor with ev3dev::motor parameter.
- void execute () override

    Perform `set_speed_regulation_enabled()` on Motor.

Additional Inherited Members

4.22.1 Detailed Description

Calls `set_speed_regulation_enabled()` method of containing Motor.

4.22.2 Constructor & Destructor Documentation

4.22.2.1 CommandMotorSetSpeedRegEnabled::CommandMotorSetSpeedRegEnabled (Motor & *motor,* bool *value*)

Constructor with ev3dev::motor parameter.

Parameters

motor	Motor to execute CommandMotor on.
value	If true, turn speed regulation on, false to turn it off.

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

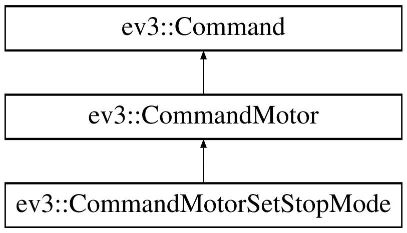
- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/CommandMotor.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/CommandMotor.cpp

4.23 ev3::CommandMotorSetStopMode Class Reference

Calls `set_stop_command()` method of containing Motor.

`#include <CommandMotor.h>`

Inheritance diagram for ev3::CommandMotorSetStopMode:



Public Types

- enum StopMode { COAST, BRAKE, HOLD }

  Stop modes for motors.

Public Member Functions

- CommandMotorSetStopMode (Motor &motor, StopMode mode)

  Constructor with ev3dev::motor parameter.
- void execute () override

  Perform `set_stop_command()` method on Motor.

Additional Inherited Members

4.23.1 Detailed Description

Calls `set_stop_command()` method of containing Motor.

4.23.2 Member Enumeration Documentation

4.23.2.1 enum **ev3::CommandMotorSetStopMode::StopMode**

Stop modes for motors.

Enumerator

COAST	No voltage. Motor slowly stops.
BRAKE	Passive braking. Motor stops faster.
HOLD	Active braking. Hardly prevent motor from any movement.

4.23.3 Constructor & Destructor Documentation

4.23.3.1 CommandMotorSetStopMode::CommandMotorSetStopMode (**Motor** & *motor,* **StopMode** *mode*)

Constructor with ev3dev::motor parameter.

Parameters

motor	Motor to execute CommandMotor on.
mode	Stop mode chosen from StopMode.

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

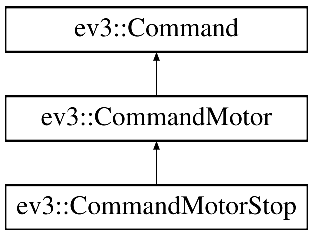
- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/CommandMotor.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/CommandMotor.cpp

4.24 ev3::CommandMotorStop Class Reference

Calls `stop()` method of containing Motor.

`#include <CommandMotor.h>`

Inheritance diagram for ev3::CommandMotorStop:



Public Member Functions

- CommandMotorStop (Motor &motor)

  Constructor with ev3dev::motor parameter.
- void execute () override

  Perform `stop()` method on Motor.

Additional Inherited Members

4.24.1 Detailed Description

Calls `stop()` method of containing Motor.

4.24.2 Constructor & Destructor Documentation

4.24.2.1 CommandMotorStop::CommandMotorStop (**Motor &** *motor*)

Constructor with ev3dev::motor parameter.

Parameters

motor	Motor to execute CommandMotor on.

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

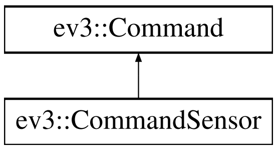
- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/CommandMotor.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/CommandMotor.cpp

4.25 ev3::CommandSensor Class Reference

Base class for all sensor controlling commands.

`#include <CommandSensor.h>`

Inheritance diagram for ev3::CommandSensor:



Public Member Functions

- CommandSensor (Sensor &sensor)

  Constructor with ev3dev::sensor parameter.
- Sensor getSensor ()

  Get sensor associated with Command.

Protected Attributes

- Sensor _sensor

  Sensor on which this CommandSensor will be executed.

4.25.1 Detailed Description

Base class for all sensor controlling commands.

See also

ev3dev::sensor

4.25.2 Constructor & Destructor Documentation

4.25.2.1 CommandSensor::CommandSensor (Sensor & *sensor*)

Constructor with ev3dev::sensor parameter.

Parameters

sensor	Sensor to execute CommandSensor on.

4.25.3 Member Function Documentation

4.25.3.1 Sensor CommandSensor::getSensor ( )

Get sensor associated with Command.

Returns

Sensor class object.

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/action/CommandSensor.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/action/CommandSensor.cpp

4.26 ev3::Communication Class Reference

Encapsulates low-level communication and adds logic concerning sending and receiving Message queueing.

```
#include <Communication.h>
```

Public Member Functions

- Communication ()

  Default constructor.
- std::thread createThread (Queue< Message > ∗sendQueue, Queue< Message > ∗receiveQueue, bool isMaster=false)

  Thread creation method (insted of running Communication in the main thread).
- void run (Queue< Message > ∗sendQueue, Queue< Message > ∗receiveQueue, bool isMaster=false)

  Starts Communication procedures.

4.26.1 Detailed Description

Encapsulates low-level communication and adds logic concerning sending and receiving Message queueing.

4.26.2 Member Function Documentation

4.26.2.1 std::thread Communication::createThread (**Queue< Message > ∗** *sendQueue,* **Queue< Message > ∗** *receiveQueue,* bool *isMaster =* `false`)

Thread creation method (insted of running Communication in the main thread).

Parameters

sendQueue	Out Message queue.
receiveQueue	In Message queue.
isMaster	True if queue is synchronized with master, false otherwise.

Returns

New std::thread object with Communication class active.

4.26.2.2 void Communication::run (**Queue< Message > ∗** *sendQueue,* **Queue< Message > ∗** *receiveQueue,* bool *isMaster =* `false`)

Starts Communication procedures.

Parameters

sendQueue	Out Message queue.
receiveQueue	In Message queue.
isMaster	True if queue is synchronized with master, false otherwise.

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/communication/Communication.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/communication/Communication.cpp

4.27 ev3::CommUtils Class Reference

Responsible for low-level communication.

```
#include <CommUtils.h>
```

Classes

- struct NetworkNode

  Stores information about a particular node in the network.

Public Member Functions

- CommUtils ()

  Default constructor.
- int preparePassiveSocket (unsigned int portNumber)

  Prepares socket for transmission on given port.
- int sendMessage (unsigned int socket, unsigned int port, Message &message, std::string &proto, bool is↩ Master, unsigned int repeat=SENT_MESSAGE_COPIES)

  General method for sending messages.
- int receiveMessage (unsigned int socket, Message &message, NetworkNode &sender)

  General receive method.
- int receiveMessageDelay (unsigned int socket, Message &message, NetworkNode &sender, unsigned int msDelay=DEFAULT_RECEIVE_DELAY)

  General receive method with waiting delay.

4.27.1 Detailed Description

Responsible for low-level communication.

Uses socket API and UNIX sending and receiving methods.

4.27.2 Member Function Documentation

4.27.2.1 int CommUtils::preparePassiveSocket (unsigned int *portNumber*)

Prepares socket for transmission on given port.

Parameters

portNumber	Port number to assign socket to.

Returns

Id of the socket assigned.

4.27.2.2 int CommUtils::receiveMessage (unsigned int *socket*, Message & *message*, NetworkNode & *sender*)

General receive method.

Parameters

socket	Previously prepared socket.
message	Message reference to be set after receiving.
sender	NetworkNode to be set after receiving.

Returns

Error code or positive integer with number of bytes received.

4.27.2.3 int CommUtils::receiveMessageDelay (unsigned int *socket*, Message & *message*, NetworkNode & *sender*, unsigned int *msDelay* = `DEFAULT_RECEIVE_DELAY`)

General receive method with waiting delay.

Parameters

socket	Previously prepared socket.
message	Message reference to be set after receiving.
sender	NetworkNode to be set after receiving.
msDelay	Maximum time in milliseconds to wait for message.

Returns

Error code or positive integer with number of bytes received.

4.27.2.4 int CommUtils::sendMessage (unsigned int *socket*, unsigned int *port*, Message & *message*, std::string & *proto*, bool *isMaster*, unsigned int *repeat* = `SENT_MESSAGE_COPIES`)

General method for sending messages.

Parameters

socket	Previously prepared socket.
port	Number of port to communicate through.
message	Message to be sent.
proto	Message prototype passed to avoid its multiple encoding.
isMaster	Flag from Communication class. True if master is the sender.
repeat	Number of copies to be sent.

Returns

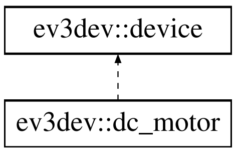
Error code or positive integer with number of bytes sent.

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/communication/CommUtils.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/communication/CommUtils.cpp

4.28 ev3dev::dc_motor Class Reference

Inheritance diagram for ev3dev::dc_motor:



Public Member Functions

- **dc_motor** (address_type address=OUTPUT_AUTO)
- auto **set_command** (std::string v) -> decltype(∗this)
- mode_set **commands** () const
- std::string **driver_name** () const
- int **duty_cycle** () const
- int **duty_cycle_sp** () const
- auto **set_duty_cycle_sp** (int v) -> decltype(∗this)
- std::string **polarity** () const
- auto **set_polarity** (std::string v) -> decltype(∗this)
- std::string **address** () const
- int **ramp_down_sp** () const
- auto **set_ramp_down_sp** (int v) -> decltype(∗this)
- int **ramp_up_sp** () const
- auto **set_ramp_up_sp** (int v) -> decltype(∗this)
- mode_set **state** () const
- auto **set_stop_command** (std::string v) -> decltype(∗this)
- mode_set **stop_commands** () const
- int **time_sp** () const
- auto **set_time_sp** (int v) -> decltype(∗this)
- void **run_forever** ()
- void **run_timed** ()
- void **run_direct** ()
- void **stop** ()

Static Public Attributes

- static const std::string **command_run_forever** { "run-forever" }
- static const std::string **command_run_timed** { "run-timed" }
- static const std::string **command_run_direct** { "run-direct" }
- static const std::string **command_stop** { "stop" }
- static const std::string **polarity_normal** { "normal" }
- static const std::string **polarity_inversed** { "inversed" }
- static const std::string **stop_command_coast** { "coast" }
- static const std::string **stop_command_brake** { "brake" }

Protected Attributes

- std::string **_port_name**

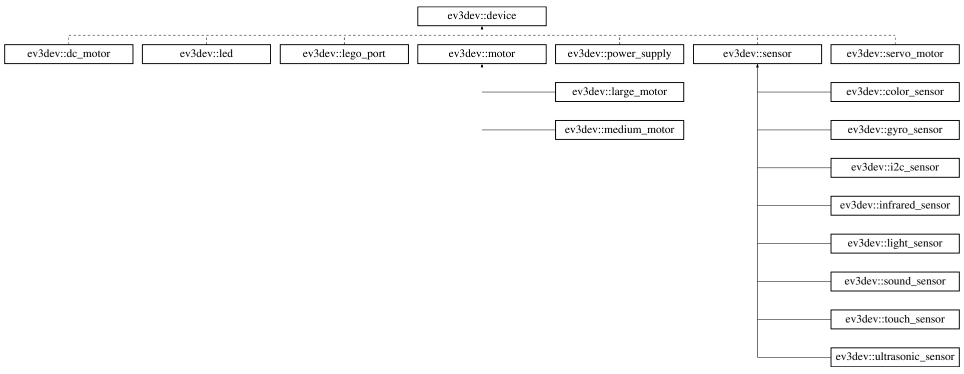
Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.29 ev3dev::device Class Reference

Inheritance diagram for ev3dev::device:



Public Member Functions

- bool **connect** (const std::string &dir, const std::string &pattern, const std::map< std::string, std::set< std↩ ::string >> &match) noexcept
- bool **connected** () const
- int **device_index** () const
- int **get_attr_int** (const std::string &name) const
- void **set_attr_int** (const std::string &name, int value)
- std::string **get_attr_string** (const std::string &name) const
- void **set_attr_string** (const std::string &name, const std::string &value)
- std::string **get_attr_line** (const std::string &name) const
- mode_set **get_attr_set** (const std::string &name, std::string ∗pCur=nullptr) const
- std::string **get_attr_from_set** (const std::string &name) const

Protected Attributes

- std::string **_path**
- int **_device_index** = -1

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.30 ev3::Devices Class Reference

Singleton class responsible for managing devices connected to the robot.

```
#include <Devices.h>
```

Public Types

- typedef std::map< ev3dev::port_type, Motor > MotorsVector

  Type for mapping Motor objects to their assigned ports.
- typedef std::map< ev3dev::port_type, Sensor > SensorsVector

  Type for mapping Sensor objects to their assigned ports.
- typedef std::vector< std::pair< ev3dev::port_type, ev3dev::device_type > > RequiredDevices

  Vector of pairs mapping port to required device.
- typedef std::map< ev3dev::port_type, SensorValue > SensorStatus

  Map containing pairs port-values for all sensors.

Public Member Functions

- bool checkDevices (RequiredDevices &devices)

  Check connected devices and requirements.
- void update ()

  Performs update on measuring values.
- void addListener (Sensor::SensorType type)

  Add listener for given Sensor type.
- void removeListener (Sensor::SensorType type)

  Remove listener for given Sensor type.
- Motor getMotor (ev3dev::port_type port)

  Motor getter.
- Sensor getSensor (ev3dev::port_type port)

  Sensor getter.
- void setSafetyTouchSensor (ev3dev::port_type port)

  Specify port on which touch sensor that detects collisions is.
- void setProximitySensor (ev3dev::port_type port)

  Specify port on which proximity sensor that detects obstacles is.
- void stopAllDevices ()

  Stops all Motors.

Static Public Member Functions

- static Devices ∗ getInstance ()

  Instance getter.
- static void destroy ()

  Deallocate instance.

Static Public Attributes

- static const ev3dev::port_type PORT_ANY {"any"}

  Can be used to define that device port is irrelevant.

Protected Member Functions

- Devices ()

  Default private constructor (preventing object construction).
- Devices (const Devices &other)

  Default private copy constructor (preventing object construction by copying).
- Devices & operator= (const Devices &other)

  Private assignment operator (preventing object assignment).
- ∼Devices ()

  Default private destructor (preventing object unwanted destruction).

Protected Attributes

- std::map< Sensor::SensorType, bool > _listeners

  Sensor listeners.
- std::map< ev3dev::port_type, int > _safetyTouchSensors

  Touch sensor for detecting collisions.
- std::map< ev3dev::port_type, int > _proximitySensors

  Proximity sensors for detecting obstacles.
- MotorsVector _motors

  Stored Motor objects.
- SensorsVector _sensors

  Stored Sensor objects.
- SensorStatus _status

  Sensors' status with all values.

Static Protected Attributes

- static Devices ∗ _instance = nullptr

  Instance of Devices singleton class.

4.30.1 Detailed Description

Singleton class responsible for managing devices connected to the robot.

4.30.2 Constructor & Destructor Documentation

4.30.2.1 ev3::Devices::Devices (const Devices & *other*) `[protected]`

Default private copy constructor (preventing object construction by copying).

Parameters

other	Other Devices object.

4.30.3 Member Function Documentation

4.30.3.1 void Devices::addListener (Sensor::SensorType *type*)

Add listener for given Sensor type.

Parameters

type	Type of Sensor for which value to watch.

4.30.3.2 bool Devices::checkDevices (RequiredDevices & *devices*)

Check connected devices and requirements.

Parameters

devices	Vector of required devices.

Returns

True if everything is connected properly, false otherwise.

4.30.3.3 Devices ∗ Devices::getInstance ( ) `[static]`

Instance getter.

Returns

Create previously or new instance of class Devices.

4.30.3.4 Motor Devices::getMotor (ev3dev::port_type *port*)

Motor getter.

Parameters

port	Port id on which the Motor is.

Returns

Motor object assigned to specified port.

4.30.3.5 Sensor Devices::getSensor (ev3dev::port_type *port*)

Sensor getter.

Parameters

port	Port id on which the Sensor is.

Returns

Sensor object assigned to specified port.

4.30.3.6 Devices& ev3::Devices::operator= (const Devices & *other*) `[protected]`

Private assignment operator (preventing object assignment).

Parameters

other	Other Devices object.

Returns

Copy of passed object.

4.30.3.7 void Devices::removeListener (Sensor::SensorType *type*)

Remove listener for given Sensor type.

Parameters

type	Type of Sensor for which value not to watch anymore.

4.30.3.8 void Devices::setProximitySensor (ev3dev::port_type *port*)

Specify port on which proximity sensor that detects obstacles is.

Parameters

port	Port for proximity sensor.

4.30.3.9 void Devices::setSafetyTouchSensor (ev3dev::port_type *port*)

Specify port on which touch sensor that detects collisions is.

Parameters

port	Port for safety touch sensor.

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

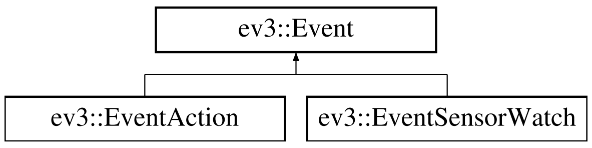
- /home/panda/Dokumenty/Repos/Ev3Dev/include/robot/Devices.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/robot/Devices.cpp

4.31 ev3::Event Class Reference

Base class for all Event classes.

```
#include <Event.h>
```

Inheritance diagram for ev3::Event:



Public Types

- enum EventType {
  EMPTY, BEHAVIOUR_START, BEHAVIOUR_STOP, SENSOR_WATCH,
  OBSTACLE_DETECTED, PROXIMITY_ALERT, ACTION_FINISHED, ACTION_INTERR }

  Event type.

Public Member Functions

- Event ()

  Default constructor.
- Event (EventType type)

  Constructor with Event type parameter.
- EventType getType ()

  Event type getter.
- std::string getStringType ()

  Get human-readable Event name.

4.31.1 Detailed Description

Base class for all Event classes.

Triggered when certain events occur during the robot's main loop execution.

4.31.2 Member Enumeration Documentation

4.31.2.1 enum **ev3::Event::EventType**

Event type.

Enumerator

EMPTY	Empty event, no meaning.
BEHAVIOUR_START	Behaviour was started.
BEHAVIOUR_STOP	Behaviour was stopped.
SENSOR_WATCH	Value was measured from sensor.
OBSTACLE_DETECTED	Robot hit an obstacle.
PROXIMITY_ALERT	Distance sensor triggered alert.
ACTION_FINISHED	Triggered when action was properly executed.
ACTION_INTERR	Triggered when action was interrupted.

4.31.3 Constructor & Destructor Documentation

4.31.3.1 Event::Event (**EventType** ***type***)

Constructor with Event type parameter.

Parameters

type	Type of the event triggered.

4.31.4 Member Function Documentation

4.31.4.1 std::string Event::getStringType ( )

Get human-readable Event name.

Returns

String with Event name.

4.31.4.2 Event::EventType Event::getType ( )

Event type getter.

Returns

EventType value.

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

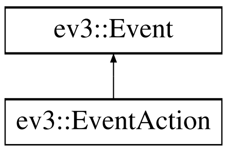
- /home/panda/Dokumenty/Repos/Ev3Dev/include/communication/Event.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/communication/Event.cpp

4.32 ev3::EventAction Class Reference

Event class triggered when something happened with Action.

```
#include <Event.h>
```

Inheritance diagram for ev3::EventAction:



Public Member Functions

- EventAction (EventType eventType, Action::ActionType actionType)

  Constructor with Event type and Action type.
- Action::ActionType getActionType ()

  Action type getter.

Additional Inherited Members

4.32.1 Detailed Description

Event class triggered when something happened with Action.

4.32.2 Constructor & Destructor Documentation

4.32.2.1 EventAction::EventAction (EventType *eventType*, Action::ActionType *actionType*)

Constructor with Event type and Action type.

Parameters

eventType	One of Event types concerning actions.
actionType	Type of Action this event concerns.

4.32.3 Member Function Documentation

4.32.3.1 Action::ActionType EventAction::getActionType ( )

Action type getter.

Returns

Stored type of Action.

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/communication/Event.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/communication/Event.cpp

4.33 ev3::EventQueue Class Reference

Public Member Functions

- void **push** (SharedPtrEvent message)
- SharedPtrEvent **pop** ()
- bool **empty** ()
- unsigned int **size** ()

Static Public Member Functions

- static EventQueue $*$ **getInstance** ()
- static void **destroy** ()

Protected Member Functions

- **EventQueue** (const EventQueue &)
- EventQueue & **operator=** (const EventQueue &)

Protected Attributes

- std::queue< SharedPtrEvent > **_queue**
- std::mutex **_mutex**

Static Protected Attributes

- static EventQueue ∗ **_instance** = nullptr

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

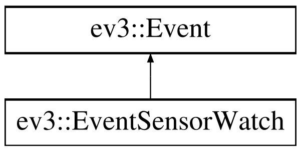
- /home/panda/Dokumenty/Repos/Ev3Dev/include/utils/EventQueue.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/utils/EventQueue.cpp

4.34 ev3::EventSensorWatch Class Reference

Triggered when measurement of certain Sensor occured.

```
#include <Event.h>
```

Inheritance diagram for ev3::EventSensorWatch:



Public Member Functions

- EventSensorWatch (Sensor::SensorType type, SensorValue value)
  Constructor with sensor type and measured value.
- SensorValue getValue ()
  Stored sensor value getter.
- Sensor::SensorType getType ()
  Stored Sensor type getter.

Additional Inherited Members

4.34.1 Detailed Description

Triggered when measurement of certain Sensor occured.

4.34.2 Constructor & Destructor Documentation

4.34.2.1 EventSensorWatch::EventSensorWatch (Sensor::SensorType *type,* SensorValue *value*)

Constructor with sensor type and measured value.

Parameters

type	Value identifying sensor type.
value	Vector with all measurements.

4.34.3 Member Function Documentation

4.34.3.1 Sensor::SensorType EventSensorWatch::getType ( )

Stored Sensor type getter.

Returns

Sensor type value.

4.34.3.2 SensorValue EventSensorWatch::getValue ( )

Stored sensor value getter.

Returns

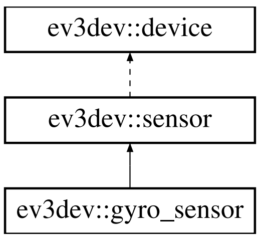
Vector with certain Sensor measurements.

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/communication/Event.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/communication/Event.cpp

4.35 ev3dev::gyro_sensor Class Reference

Inheritance diagram for ev3dev::gyro_sensor:



Public Member Functions

- **gyro_sensor** (address_type address=INPUT_AUTO)
- int **angle** ()
- int **rate** ()

Static Public Attributes

- static const std::string **mode_gyro_ang** { "GYRO-ANG" }
- static const std::string **mode_gyro_rate** { "GYRO-RATE" }
- static const std::string **mode_gyro_fas** { "GYRO-FAS" }
- static const std::string **mode_gyro_g_a** { "GYRO-G&A" }
- static const std::string **mode_gyro_cal** { "GYRO-CAL" }

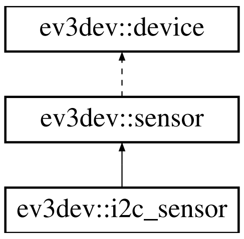
Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.36 ev3dev::i2c_sensor Class Reference

Inheritance diagram for ev3dev::i2c_sensor:



Public Member Functions

- **i2c_sensor** (address_type address=INPUT_AUTO)
- std::string **fw_version** () const
- int **poll_ms** () const
- auto **set_poll_ms** (int v) -> decltype(∗this)

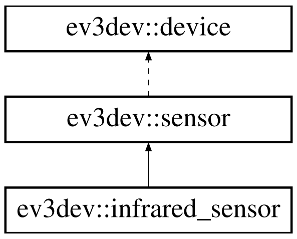
Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.37 ev3dev::infrared_sensor Class Reference

Inheritance diagram for ev3dev::infrared_sensor:



Public Member Functions

- **infrared_sensor** (address_type address=INPUT_AUTO)
- int **proximity** ()

Static Public Attributes

- static const std::string **mode_ir_prox** { "IR-PROX" }
- static const std::string **mode_ir_seek** { "IR-SEEK" }
- static const std::string **mode_ir_remote** { "IR-REMOTE" }
- static const std::string **mode_ir_rem_a** { "IR-REM-A" }
- static const std::string **mode_ir_cal** { "IR-CAL" }

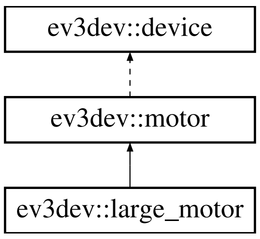
Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.38 ev3dev::large_motor Class Reference

Inheritance diagram for ev3dev::large_motor:



Public Member Functions

- **large_motor** (address_type address=OUTPUT_AUTO)

Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.39 ev3dev::lcd Class Reference

Public Member Functions

- bool **available** () const
- uint32_t **resolution_x** () const
- uint32_t **resolution_y** () const
- uint32_t **bits_per_pixel** () const
- uint32_t **frame_buffer_size** () const
- uint32_t **line_length** () const
- unsigned char ∗ **frame_buffer** ()
- void **fill** (unsigned char pixel)

Protected Member Functions

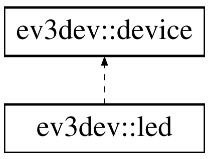
- void **init** ()
- void **deinit** ()

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.40 ev3dev::led Class Reference

Inheritance diagram for ev3dev::led:



Public Member Functions

- **led** (std::string name)
- int **max_brightness** () const
- int **brightness** () const
- auto **set_brightness** (int v) -> decltype(∗this)
- mode_set **triggers** () const
- std::string **trigger** () const
- auto **set_trigger** (std::string v) -> decltype(∗this)
- int **delay_on** () const
- auto **set_delay_on** (int v) -> decltype(∗this)
- int **delay_off** () const
- auto **set_delay_off** (int v) -> decltype(∗this)
- float **brightness_pct** () const
- auto **set_brightness_pct** (float v) -> decltype(∗this)
- void **on** ()
- void **off** ()
- void **flash** (unsigned on_ms, unsigned off_ms)

Static Public Member Functions

- static void **set_color** (const std::vector< led ∗ > &group, const std::vector< float > &color)
- static void **all_off** ()

Static Public Attributes

- static led **red_left** {"ev3:left:red:ev3dev"}
- static led **red_right** {"ev3:right:red:ev3dev"}
- static led **green_left** {"ev3:left:green:ev3dev"}
- static led **green_right** {"ev3:right:green:ev3dev"}
- static std::vector< led ∗ > **left** { &led::red_left, &led::green_left }
- static std::vector< led ∗ > **right** { &led::red_right, &led::green_right }
- static std::vector< float > **red** { static_cast<float>(1), static_cast<float>(0) }
- static std::vector< float > **green** { static_cast<float>(0), static_cast<float>(1) }
- static std::vector< float > **amber** { static_cast<float>(1), static_cast<float>(1) }
- static std::vector< float > **orange** { static_cast<float>(1), static_cast<float>(0.5) }
- static std::vector< float > **yellow** { static_cast<float>(0.5), static_cast<float>(1) }

Protected Attributes

- int **_max_brightness** = 0

Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.41 ev3::LedControl Class Reference

Class specifically designed to eliminate ev3dev library limitations of controlling LED panel.

```
#include <LedControl.h>
```

Public Types

- enum LedType {
  RED_L = 1, RED_R = 1 << 1, GREEN_L = 1 << 2, GREEN_R = 1 << 3,
  RED_ALL = RED_L | RED_R, GREEN_ALL = GREEN_L | GREEN_R, ALL = RED_ALL | GREEN_ALL }

  Type of LED diode.
- enum LedColors { RED, AMBER, YELLOW, GREEN }

  Predefined colors, that particular combination of diodes can represent.

Public Member Functions

- virtual ∼LedControl ()

  Default destructor.
- void on (unsigned int leds=LedType::ALL, unsigned int brightness=MAX_BRIGHTNESS)

  Turn the specified diodes on.
- void onExclusive (unsigned int leds=LedType::ALL, unsigned int brightness=MAX_BRIGHTNESS)

  Turn the specified diodes on and also turn off the other ones.
- void off (unsigned int leds=LedType::ALL)

  Turn the specified diodes off.
- void setColor (LedColors color)

  Set diodes to match particular color.
- void reset ()

  Ends flashing and turns all diodes off.
- void flash (unsigned int leds, unsigned int msInterval, unsigned int repeat=1, unsigned int brightnessRed=M↩
  AX_BRIGHTNESS, unsigned int brightnessGreen=MAX_BRIGHTNESS)

  Orders diodes to flash with given interval.
- void flashColor (LedColors color, unsigned int msInterval, unsigned int repeat=1)

  Orders dioded to flash a particular color with given interval.
- void endFlashing ()

  Stops flashing.

Static Public Attributes

- static const unsigned int MAX_BRIGHTNESS = 255

  Maximum value of brightness.

4.41.1 Detailed Description

Class specifically designed to eliminate ev3dev library limitations of controlling LED panel.

4.41.2 Member Enumeration Documentation

4.41.2.1 enum ev3::LedControl::LedColors

Predefined colors, that particular combination of diodes can represent.

Enumerator

RED	Only red diode.
AMBER	Red with a little bit of green.
YELLOW	Little red and full green.
GREEN	Only green diode.

4.41.2.2 enum ev3::LedControl::LedType

Type of LED diode.

Enumerator

RED_L	Red left diode.
RED_R	Red right diode.
GREEN_L	Green left diode.
GREEN_R	Green right diode.
RED_ALL	Both red diodes.
GREEN_ALL	Both green diodes.
ALL	All four diodes.

4.41.3 Member Function Documentation

4.41.3.1 void LedControl::flash (unsigned int *leds,* unsigned int *msInterval,* unsigned int *repeat* = `1`, unsigned int *brightnessRed* = MAX_BRIGHTNESS, unsigned int *brightnessGreen* = MAX_BRIGHTNESS)

Orders diodes to flash with given interval.

Parameters

leds	Combination of LedControl::LedType values.
msInterval	Flash interval in milliseconds.
repeat	Number of iterations or 0 for infinite flashing.
brightnessRed	Brightness of the red diodes.
brightnessGreen	Brightness of the green diodes.

4.41.3.2 void LedControl::flashColor (LedColors *color,* unsigned int *msInterval,* unsigned int *repeat* = `1`)

Orders dioded to flash a particular color with given interval.

Parameters

color	Type of color to be displayed.
msInterval	Flash interval in milliseconds.
repeat	Number of iterations or 0 for infinite flashing.

4.41.3.3 void LedControl::off (unsigned int *leds* = `LedType::ALL`)

Turn the specified diodes off.

Parameters

leds	Combination of LedControl::LedType values.

4.41.3.4 void LedControl::on (unsigned int *leds* = `LedType::ALL`, unsigned int *brightness* = MAX_BRIGHTNESS)

Turn the specified diodes on.

Parameters

leds	Combination of LedControl::LedType values.
brightness	Value of brightness to be set.

4.41.3.5 void LedControl::onExclusive (unsigned int *leds* = `LedType::ALL`, unsigned int *brightness* = MAX_BRIGHTNESS)

Turn the specified diodes on and also turn off the other ones.

Parameters

leds	Combination of LedControl::LedType values.
brightness	Value of brightness to be set.

4.41.3.6 void LedControl::setColor (LedColors *color*)

Set diodes to match particular color.

Parameters

color	Type of to be displayed.

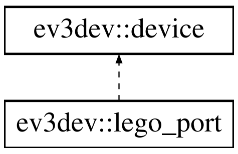
The documentation for this class was generated from the following files:

- /home/panda/Dokumenty/Repos/Ev3Dev/include/control/LedControl.h

- /home/panda/Dokumenty/Repos/Ev3Dev/src/control/LedControl.cpp

4.42 ev3dev::lego_port Class Reference

Inheritance diagram for ev3dev::lego_port:



Public Member Functions

- **lego_port** (address_type)
- std::string **driver_name** () const
- mode_set **modes** () const
- std::string **mode** () const
- auto **set_mode** (std::string v) -> decltype(∗this)
- std::string **address** () const
- auto **set_set_device** (std::string v) -> decltype(∗this)
- std::string **status** () const

Protected Member Functions

- bool **connect** (const std::map< std::string, std::set< std::string >> &) noexcept

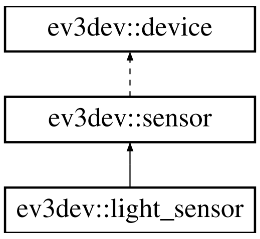
Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.43 ev3dev::light_sensor Class Reference

Inheritance diagram for ev3dev::light_sensor:



Public Member Functions

- **light_sensor** (address_type address=INPUT_AUTO)
- float **reflected_light_intensity** ()
- float **ambient_light_intensity** ()

Static Public Attributes

- static const std::string **mode_reflect** { "REFLECT" }
- static const std::string **mode_ambient** { "AMBIENT" }

Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.44 ev3::Logger Class Reference

Public Types

- enum **LogLevel** {
  DEBUG = 1, **VERBOSE** = 1 $<<$ 1, **INFO** = 1 $<<$ 2, **WARNING** = 1 $<<$ 3,
  ERROR = 1 $<<$ 4 }
- enum **LogOutput** { **STD_OUT** = 1, **STD_ERR** = 1 $<<$ 1, **FILE** = 1 $<<$ 2 }

Public Member Functions

- void **log** (std::string message, LogLevel level, LogOutput output=STD_OUT)
- void **setLogLevel** (LogLevel level)
- void **setLogLevel** (std::string level)
- void **setLogOutput** (LogOutput output)

Static Public Member Functions

- static Logger $*$ **getInstance** ()
- static void **destroy** ()

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/utils/Logger.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/utils/Logger.cpp

4.45 ev3::Master Class Reference

Controls the whole system and knows about every Agent.

```
#include <Master.h>
```

Public Types

- typedef std::map< unsigned int, Agent > AgentMap

  Type for mapping Agents to their ids.

Public Member Functions

- std::thread createThread (Queue< Message > ∗sendQueue, Queue< Message > ∗receiveQueue)

  Creates thread instead of running Master in the main thread.
- void run (Queue< Message > ∗sendQueue, Queue< Message > ∗receiveQueue)

  Starts Master procedures.
- void send (Message message, bool recordMessage=true)

  Sending method assigning id to the message.
- void stop ()

  Stop Master main loop and exit.

4.45.1 Detailed Description

Controls the whole system and knows about every Agent.

Initiates Behaviour and receives values from sensor.

4.45.2 Member Function Documentation

4.45.2.1 std::thread Master::createThread (**Queue< Message > ∗** ***sendQueue,*** **Queue< Message > ∗** ***receiveQueue***)

Creates thread instead of running Master in the main thread.

Parameters

sendQueue	Out Message queue.
receiveQueue	In Message queue.

Returns

New std::thread object with active Master class.

4.45.2.2 void Master::run (**Queue< Message > ∗** ***sendQueue,*** **Queue< Message > ∗** ***receiveQueue***)

Starts Master procedures.

Parameters

sendQueue	
receiveQueue	

4.45.2.3 void Master::send (Message *message,* bool *recordMessage* = `true`)

Sending method assigning id to the message.

Parameters

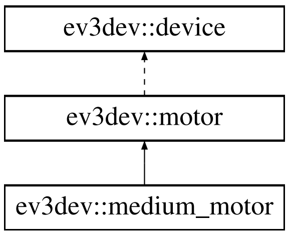
message	Message to be passed to Communication thread via sendQueue.
recordMessage	True if information about message should be saved for further purposes, false otherwise.

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/master/Master.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/master/Master.cpp

4.46 ev3dev::medium_motor Class Reference

Inheritance diagram for ev3dev::medium_motor:



Public Member Functions

- **medium_motor** (address_type address=OUTPUT_AUTO)

Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.47 ev3::Message Class Reference

Stores information passed between physical system units (another robots or master).

```
#include <Message.h>
```

Public Types

- enum MessageType {
  EMPTY, ACK, NOT, AGENT,
  MASTER, MASTER_OVER, PING, PONG,
  AGENT_OVER, ABORT, BEHAVIOUR, START,
  RESUME, PAUSE, ACTION_OK, ACTION_INTERR,
  SENSOR_VALUE, MEASURE }

  Messge Type.

Public Member Functions

- Message ()

  Default constructor.
- Message (unsigned int senderId, unsigned int receiverId, unsigned int messageId, MessageType type, StringVector parameters={})

  Full message constructor.
- unsigned int getSenderId ()

  Sender id getter.
- unsigned int getReceiverId ()

  Receiver id getter.
- unsigned int getMessageId ()

  Consequently incremented integer id getter.
- MessageType getType ()

  Message type getter.
- StringVector getParameters ()

  Message parameters getter.
- void setSenderId (unsigned int id)

  Sender id setter.
- void setReceiverId (unsigned int id)

  Receiver id setter.
- void setMessageId (unsigned int id)

  Consequently incremented integer id setter.
- void setType (MessageType type)

  Message type setter.
- void setParameters (StringVector parameters)

  Message parameters setter.
- bool empty ()

  Tell whether Message type is EMPTY.
- std::string getString ()

  Human-readable name getter.
- void reset ()

  Reset all values to default ones and type to EMPTY.

Static Public Member Functions

- static std::string encodeMessage (Message &message)

  Encode message data into string.
- static Message decodeMessage (const std::string message)

  Decode string into Message object.

4.47.1 Detailed Description

Stores information passed between physical system units (another robots or master).

4.47.2 Member Enumeration Documentation

4.47.2.1 enum ev3::Message::MessageType

Messge Type.

Enumerator

EMPTY	Empty message, no meaning.
ACK	Accept previously received request.
NOT	Deny previously received request.
AGENT	Agent side synchronization.
MASTER	Master side synchronization.
MASTER_OVER	Master work finished.
PING	Connection sustain request.
PONG	Connection sustain answer.
AGENT_OVER	Agent work finished.
ABORT	Exit processing now.
BEHAVIOUR	Behaviour definition received.
START	Behaviour start.
RESUME	Behaviour resume.
PAUSE	Behaviour pause.
ACTION_OK	Action finished correctly.
ACTION_INTERR	Action interrupted.
SENSOR_VALUE	Sensor measurement occured.
MEASURE	Instructions what to measure.

4.47.3 Constructor & Destructor Documentation

4.47.3.1 Message::Message (unsigned int *senderId,* unsigned int *receiverId,* unsigned int *messageId,* MessageType *type,* StringVector *parameters* = { })

Full message constructor.

Parameters

senderId	Id of the sender (given by master).
receiverId	Id of the receiver.
message↩ Id	Consequently incremented message id.
type	Predefined Message type.
parameters	Vector of additional, optional string parameters.

4.47.4 Member Function Documentation

4.47.4.1 Message Message::decodeMessage (const std::string *message*) [static]

Decode string into Message object.

Parameters

message	String value to be decoded.

Returns

Message object decoded, if processed successfully.

4.47.4.2 bool Message::empty ( )

Tell whether Message type is EMPTY.

Returns

True if Messge is EMPTY, false otherwise.

4.47.4.3 std::string Message::encodeMessage (Message & *message*) [static]

Encode message data into string.

Parameters

message	Reference to message object to be encoded.

Returns

String with encoded data of the message.

4.47.4.4 unsigned int Message::getMessageId ( )

Consequently incremented integer id getter.

Returns

Id of the message.

4.47.4.5 StringVector Message::getParameters ( )

Message parameters getter.

Returns

String vector with all optional parameters.

4.47.4.6 unsigned int Message::getReceiverId ( )

Receiver id getter.

Returns

Id of the message receiver.

4.47.4.7 unsigned int Message::getSenderId ( )

Sender id getter.

Returns

Id of the message sender (should be set to the value of the main class executing this method).

4.47.4.8 std::string Message::getString ( )

Human-readable name getter.

Returns

Formatted string containing name and all parameters.

4.47.4.9 Message::MessageType Message::getType ( )

Message type getter.

Returns

Enum value with Message type.

4.47.4.10 void Message::setMessageId (unsigned int *id*)

Consequently incremented integer id setter.

Parameters

id	Id of the message.

4.47.4.11 void Message::setParameters (StringVector *parameters*)

Message parameters setter.

Parameters

parameters	String vector with all optional parameters.

4.47.4.12 void Message::setReceiverId (unsigned int *id*)

Receiver id setter.

Parameters

id	Id of the message receiver.

4.47.4.13 void Message::setSenderId (unsigned int *id*)

Sender id setter.

Parameters

id	Id of the message sender (should be set to the value of the main class executing this method).

4.47.4.14 void Message::setType (MessageType *type*)

Message type setter.

Parameters

type	Enum value with Message type.

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/communication/Message.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/communication/Message.cpp

4.48 ev3::Motor Class Reference

Encapsulates ev3dev::motor.

```
#include <Motor.h>
```

Public Member Functions

- Motor (ev3dev::motor motor)

  Constructor with Motor.
- ev3dev::motor getMotor ()

  Motor getter.

4.48.1 Detailed Description

Encapsulates ev3dev::motor.

Can provide additional logic.

4.48.2 Constructor & Destructor Documentation

4.48.2.1 Motor::Motor (**ev3dev::motor** *motor*)

Constructor with Motor.

Parameters

motor	ev3dev::Motor object.

4.48.3 Member Function Documentation

4.48.3.1 **ev3dev::motor** Motor::getMotor ( )

Motor getter.

Returns

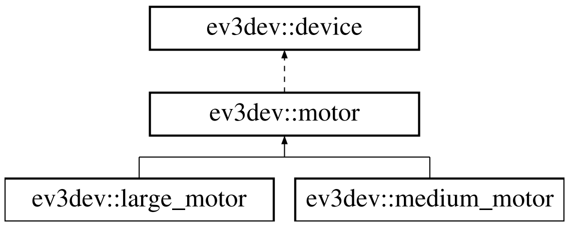
Stored ev3dev::motor object.

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/robot/Motor.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/robot/Motor.cpp

4.49 ev3dev::motor Class Reference

Inheritance diagram for ev3dev::motor:



Public Types

- typedef device_type **motor_type**

Public Member Functions

- **motor** (address_type)
- **motor** (address_type, const motor_type &)
- auto **set_command** (std::string v) -> decltype(∗this)
- mode_set **commands** () const
- int **count_per_rot** () const
- std::string **driver_name** () const
- int **duty_cycle** () const
- int **duty_cycle_sp** () const
- auto **set_duty_cycle_sp** (int v) -> decltype(∗this)
- std::string **encoder_polarity** () const
- auto **set_encoder_polarity** (std::string v) -> decltype(∗this)
- std::string **polarity** () const
- auto **set_polarity** (std::string v) -> decltype(∗this)
- std::string **address** () const
- int **position** () const
- auto **set_position** (int v) -> decltype(∗this)
- int **position_p** () const
- auto **set_position_p** (int v) -> decltype(∗this)
- int **position_i** () const
- auto **set_position_i** (int v) -> decltype(∗this)
- int **position_d** () const
- auto **set_position_d** (int v) -> decltype(∗this)
- int **position_sp** () const
- auto **set_position_sp** (int v) -> decltype(∗this)
- int **speed** () const
- int **speed_sp** () const
- auto **set_speed_sp** (int v) -> decltype(∗this)
- int **ramp_up_sp** () const
- auto **set_ramp_up_sp** (int v) -> decltype(∗this)
- int **ramp_down_sp** () const
- auto **set_ramp_down_sp** (int v) -> decltype(∗this)
- std::string **speed_regulation_enabled** () const
- auto **set_speed_regulation_enabled** (std::string v) -> decltype(∗this)

- int **speed_regulation_p** () const
- auto **set_speed_regulation_p** (int v) -> decltype(∗this)
- int **speed_regulation_i** () const
- auto **set_speed_regulation_i** (int v) -> decltype(∗this)
- int **speed_regulation_d** () const
- auto **set_speed_regulation_d** (int v) -> decltype(∗this)
- mode_set **state** () const
- std::string **stop_command** () const
- auto **set_stop_command** (std::string v) -> decltype(∗this)
- mode_set **stop_commands** () const
- int **time_sp** () const
- auto **set_time_sp** (int v) -> decltype(∗this)
- void **run_forever** ()
- void **run_to_abs_pos** ()
- void **run_to_rel_pos** ()
- void **run_timed** ()
- void **run_direct** ()
- void **stop** ()
- void **reset** ()
- motor_type **type_name** ()

Static Public Attributes

- static const motor_type **motor_large** { "lego-ev3-l-motor" }
- static const motor_type **motor_medium** { "lego-ev3-m-motor" }
- static const std::string **command_run_forever** { "run-forever" }
- static const std::string **command_run_to_abs_pos** { "run-to-abs-pos" }
- static const std::string **command_run_to_rel_pos** { "run-to-rel-pos" }
- static const std::string **command_run_timed** { "run-timed" }
- static const std::string **command_run_direct** { "run-direct" }
- static const std::string **command_stop** { "stop" }
- static const std::string **command_reset** { "reset" }
- static const std::string **encoder_polarity_normal** { "normal" }
- static const std::string **encoder_polarity_inversed** { "inversed" }
- static const std::string **polarity_normal** { "normal" }
- static const std::string **polarity_inversed** { "inversed" }
- static const std::string **speed_regulation_on** { "on" }
- static const std::string **speed_regulation_off** { "off" }
- static const std::string **stop_command_coast** { "coast" }
- static const std::string **stop_command_brake** { "brake" }
- static const std::string **stop_command_hold** { "hold" }

Protected Member Functions

- bool **connect** (const std::map< std::string, std::set< std::string >> &) noexcept

Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.50 ev3::CommUtils::NetworkNode Struct Reference

Stores information about a particular node in the network.

```
#include <CommUtils.h>
```

Public Attributes

- unsigned int port

  Port number.
- std::string ipAddress

  Node's ipv4 address.

4.50.1 Detailed Description

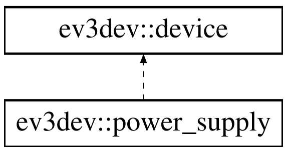
Stores information about a particular node in the network.

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/communication/CommUtils.h

4.51 ev3dev::power_supply Class Reference

Inheritance diagram for ev3dev::power_supply:



Public Member Functions

- **power_supply** (std::string name)
- int **measured_current** () const
- int **measured_voltage** () const
- int **max_voltage** () const
- int **min_voltage** () const
- std::string **technology** () const
- std::string **type** () const
- float **measured_amps** () const
- float **measured_volts** () const

Static Public Attributes

- static power_supply **battery** { "" }

Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.52 ev3::Queue< T > Class Template Reference

Public Member Functions

- void **push** (T message)
- T **pop** ()
- bool **empty** ()

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/utils/Queue.h

4.53 ev3dev::remote_control Class Reference

Public Types

- enum **buttons** {
  red_up = (1 << 0), **red_down** = (1 << 1), **blue_up** = (1 << 2), **blue_down** = (1 << 3),
  beacon = (1 << 4) }

Public Member Functions

- **remote_control** (unsigned channel=1)
- **remote_control** (infrared_sensor &, unsigned channel=1)
- bool **connected** () const
- unsigned **channel** () const
- bool **process** ()

Public Attributes

- std::function< void(bool)> **on_red_up**
- std::function< void(bool)> **on_red_down**
- std::function< void(bool)> **on_blue_up**
- std::function< void(bool)> **on_blue_down**
- std::function< void(bool)> **on_beacon**
- std::function< void(int)> **on_state_change**

Protected Member Functions

- virtual void **on_value_changed** (int value)

Protected Attributes

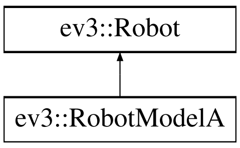
- infrared_sensor ∗ **_sensor** = nullptr
- bool **_owns_sensor** = false
- unsigned **_channel** = 0
- int **_value** = 0
- int **_state** = 0

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.54 ev3::Robot Class Reference

Inheritance diagram for ev3::Robot:



Public Types

- typedef std::vector< Action::ActionType > AvailableActions

  Type for specifying all available actions for given Robot model.

Public Member Functions

- **Robot** (Devices::RequiredDevices devices, AvailableActions actions)
- std::thread **createThread** (Queue< Message > ∗sendQueue, Queue< Message > ∗receiveQueue)
- virtual void **run** (Queue< Message > ∗sendQueue, Queue< Message > ∗receiveQueue)
- void **stop** ()
- void **send** (Message message)
- virtual std::string **getString** ()

Protected Member Functions

- virtual SharedPtrBehaviour **generateBehaviour** (Behaviour::BehaviourType type, StringVector parameters)

Protected Attributes

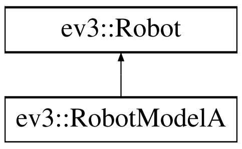
- unsigned int **_id** = 0
- unsigned int **_commId** = 0
- float **_pulsePerUnitRatio** = 1.f
- Devices::RequiredDevices **_requiredDevices**
- AvailableActions **_availableActions**
- Queue< Message > ∗ **_sendQueue**
- Queue< Message > ∗ **_receiveQueue**
- LedControl **_ledControl**
- RobotState ∗ **_state** = new RobotStateIdle(&_ledControl)

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/robot/Robot.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/robot/Robot.cpp

4.55 ev3::RobotModelA Class Reference

Inheritance diagram for ev3::RobotModelA:



Public Member Functions

- virtual std::string **getString** () override

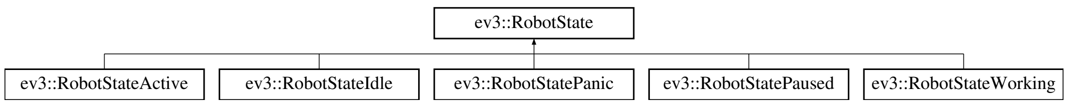
Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/robot/RobotModelA.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/robot/RobotModelA.cpp

4.56 ev3::RobotState Class Reference

Inheritance diagram for ev3::RobotState:



Public Types

- enum **States** {
  IDLE, **ACTIVE**, **WORKING**, **PAUSED**,
  PANIC }
- typedef std::map< Message::MessageType, States > **ChangeMap**

Public Member Functions

- **RobotState** (ChangeMap changes, LedControl ∗led)
- virtual RobotState ∗ **process** (Message msg)
- Message::MessageType **getPendingMessage** ()
- void **updateTimer** ()
- bool **isPendingEnabled** ()
- void **setBehaviour** (SharedPtrBehaviour behaviour)
- SharedPtrBehaviour **getBehaviour** ()

Static Public Attributes

- static const float **MASTER_TIMEOUT** = 10.f ∗ 1000
- static const float **MASTER_PING_TIME** = 3.f ∗ 1000

Protected Member Functions

- RobotState ∗ **switchState** (Message::MessageType type)
- RobotState ∗ **changeState** (States state)

Protected Attributes

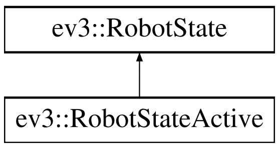
- SharedPtrBehaviour **_currentBehaviour**
- States **_state**
- ChangeMap **_changes**
- LedControl ∗ **_led**
- Message::MessageType **_pendingMessage** = Message::EMPTY
- float **_pendingTimeout** = 0.f
- HighResClock::time_point **_masterTimeout** = HighResClock::now()
- HighResClock::time_point **_messageDelay** = HighResClock::now()

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/robot/RobotState.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/robot/RobotState.cpp

4.57 ev3::RobotStateActive Class Reference

Inheritance diagram for ev3::RobotStateActive:



Public Member Functions

- **RobotStateActive** (LedControl ∗led)
- RobotState ∗ **process** (Message msg)

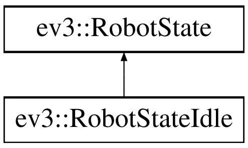
Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/robot/RobotState.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/robot/RobotState.cpp

4.58 ev3::RobotStateIdle Class Reference

Inheritance diagram for ev3::RobotStateIdle:



Public Member Functions

- **RobotStateIdle** (LedControl ∗led)
- RobotState ∗ **process** (Message msg)

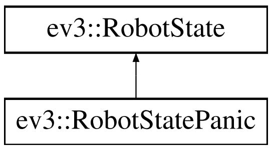
Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/robot/RobotState.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/robot/RobotState.cpp

4.59 ev3::RobotStatePanic Class Reference

Inheritance diagram for ev3::RobotStatePanic:



Public Member Functions

- **RobotStatePanic** (LedControl ∗led)
- RobotState ∗ **process** (Message msg)

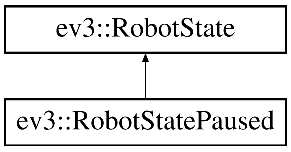
Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/robot/RobotState.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/robot/RobotState.cpp

4.60 ev3::RobotStatePaused Class Reference

Inheritance diagram for ev3::RobotStatePaused:



Public Member Functions

- **RobotStatePaused** (LedControl ∗led)
- RobotState ∗ **process** (Message msg)

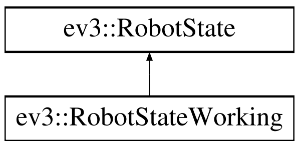
Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/robot/RobotState.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/robot/RobotState.cpp

4.61 ev3::RobotStateWorking Class Reference

Inheritance diagram for ev3::RobotStateWorking:



Public Member Functions

- **RobotStateWorking** (LedControl ∗led)
- RobotState ∗ **process** (Message msg)

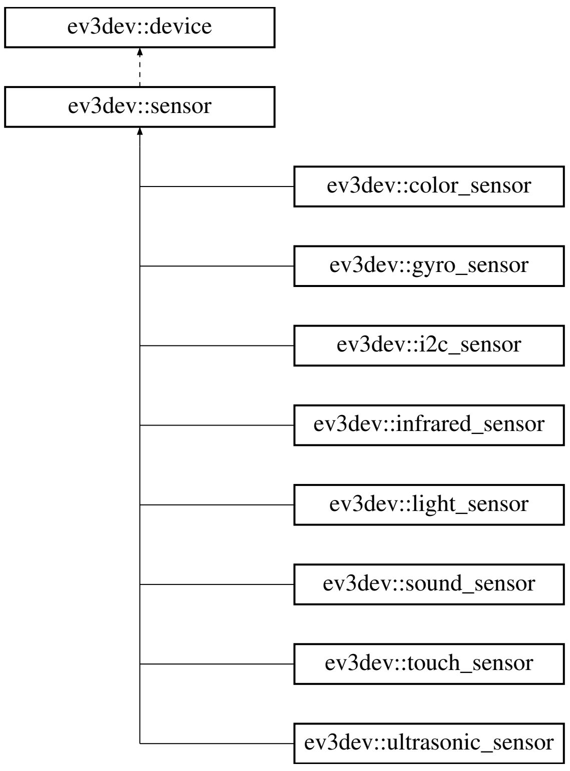
Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/robot/RobotState.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/robot/RobotState.cpp

4.62 ev3dev::sensor Class Reference

Inheritance diagram for ev3dev::sensor:



Public Types

- typedef device_type **sensor_type**

Public Member Functions

- **sensor** (address_type)
- **sensor** (address_type, const std::set< sensor_type > &)
- int **value** (unsigned index=0) const
- float **float_value** (unsigned index=0) const
- std::string **type_name** () const
- std::string **bin_data_format** () const
- const std::vector< char > & **bin_data** () const
- template<class T >
  void **bin_data** (T *buf) const
- auto **set_command** (std::string v) -> decltype(*this)
- mode_set **commands** () const
- int **decimals** () const
- std::string **driver_name** () const
- std::string **mode** () const
- auto **set_mode** (std::string v) -> decltype(*this)
- mode_set **modes** () const
- int **num_values** () const
- std::string **address** () const
- std::string **units** () const

Static Public Attributes

- static const sensor_type **ev3_touch** { "lego-ev3-touch" }
- static const sensor_type **ev3_color** { "lego-ev3-color" }
- static const sensor_type **ev3_ultrasonic** { "lego-ev3-us" }
- static const sensor_type **ev3_gyro** { "lego-ev3-gyro" }
- static const sensor_type **ev3_infrared** { "lego-ev3-ir" }
- static const sensor_type **nxt_touch** { "lego-nxt-touch" }
- static const sensor_type **nxt_light** { "lego-nxt-light" }
- static const sensor_type **nxt_sound** { "lego-nxt-sound" }
- static const sensor_type **nxt_ultrasonic** { "lego-nxt-us" }
- static const sensor_type **nxt_i2c_sensor** { "nxt-i2c-sensor" }
- static const sensor_type **nxt_analog** { "nxt-analog" }

Protected Member Functions

- bool **connect** (const std::map< std::string, std::set< std::string >> &) noexcept

Protected Attributes

- std::vector< char > **_bin_data**

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.63 ev3::Sensor Class Reference

Encapsulates ev3dev::sensor.

```
#include <Sensor.h>
```

Public Types

- enum SensorType {
  TOUCH, COLOR, ULTRASONIC, GYRO,
  INFRARED, SOUND, LIGHT }

  Sensor type.

Public Member Functions

- Sensor (ev3dev::sensor sensor, SensorType type)
- ev3dev::sensor getSensor ()
- int getValue (unsigned int n)
- float getValueF (unsigned int n)
- int getDecimals ()
- unsigned int getNumValues ()
- SensorType getType ()

Static Public Member Functions

- static StringVector prepareMessage (SensorValue value, SensorType type)

4.63.1 Detailed Description

Encapsulates ev3dev::sensor.

Can provide additional logic.

4.63.2 Member Enumeration Documentation

4.63.2.1 enum ev3::Sensor::SensorType

Sensor type.

Enumerator

TOUCH	Touch sensor.
COLOR	Color sensor.
ULTRASONIC	Ultrasonic sensor.
GYRO	Gyroscope sensor.
INFRARED	Infrared sensor.
SOUND	Sound sensor.
LIGHT	Light sensor.

4.63.3 Constructor & Destructor Documentation

4.63.3.1 Sensor::Sensor (**ev3dev::sensor** *sensor,* **SensorType** *type*)

Parameters

sensor	
type	

4.63.4 Member Function Documentation

4.63.4.1 int Sensor::getDecimals ( )

Returns

4.63.4.2 unsigned int Sensor::getNumValues ( )

Returns

4.63.4.3 ev3dev::sensor Sensor::getSensor ( )

Returns

4.63.4.4 Sensor::SensorType Sensor::getType ( )

Returns

4.63.4.5 int Sensor::getValue (unsigned int *n*)

Parameters

n	

Returns

4.63.4.6 float Sensor::getValueF (unsigned int *n*)

Parameters

n	

Returns

4.63.4.7 StringVector Sensor::prepareMessage (SensorValue *value*, SensorType *type*) [static]

Parameters

value	
type	

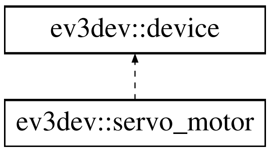
Returns

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/robot/Sensor.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/robot/Sensor.cpp

4.64 ev3dev::servo_motor Class Reference

Inheritance diagram for ev3dev::servo_motor:



Public Member Functions

- **servo_motor** (address_type address=OUTPUT_AUTO)
- auto **set_command** (std::string v) -> decltype(∗this)
- std::string **driver_name** () const
- int **max_pulse_sp** () const
- auto **set_max_pulse_sp** (int v) -> decltype(∗this)
- int **mid_pulse_sp** () const
- auto **set_mid_pulse_sp** (int v) -> decltype(∗this)
- int **min_pulse_sp** () const
- auto **set_min_pulse_sp** (int v) -> decltype(∗this)
- std::string **polarity** () const
- auto **set_polarity** (std::string v) -> decltype(∗this)
- std::string **address** () const

- int **position_sp** () const
- auto **set_position_sp** (int v) -> decltype(∗this)
- int **rate_sp** () const
- auto **set_rate_sp** (int v) -> decltype(∗this)
- mode_set **state** () const
- void **run** ()
- void **float_** ()

Static Public Attributes

- static const std::string **command_run** { "run" }
- static const std::string **command_float** { "float" }
- static const std::string **polarity_normal** { "normal" }
- static const std::string **polarity_inversed** { "inversed" }

Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.65 ev3::SignalHandler Class Reference

Static Public Member Functions

- static void **HandleSignal** (int signum)

Static Public Attributes

- static Robot ∗ **robot** = nullptr
- static Master ∗ **master** = nullptr

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/utils/SignalHandler.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/utils/SignalHandler.cpp

4.66 ev3dev::sound Class Reference

Static Public Member Functions

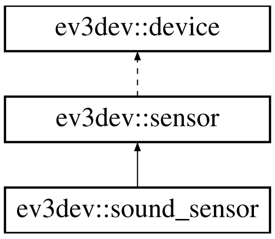
- static void **beep** (const std::string &args="", bool bSynchronous=false)
- static void **tone** (float frequency, float ms, bool bSynchronous=false)
- static void **tone** (const std::vector< std::vector< float > > &sequence, bool bSynchronous=false)
- static void **play** (const std::string &soundfile, bool bSynchronous=false)
- static void **speak** (const std::string &text, bool bSynchronous=false)

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.67 ev3dev::sound_sensor Class Reference

Inheritance diagram for ev3dev::sound_sensor:



Public Member Functions

- **sound_sensor** (address_type address=INPUT_AUTO)
- float **sound_pressure** ()
- float **sound_pressure_low** ()

Static Public Attributes

- static const std::string **mode_db** { "DB" }
- static const std::string **mode_dba** { "DBA" }

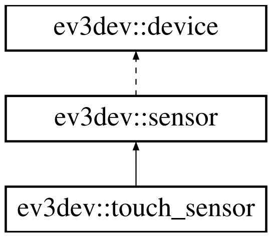
Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.68 ev3dev::touch_sensor Class Reference

Inheritance diagram for ev3dev::touch_sensor:



Public Member Functions

- **touch_sensor** (address_type address=INPUT_AUTO)
- bool **is_pressed** ()

Static Public Attributes

- static const std::string **mode_touch** { "TOUCH" }

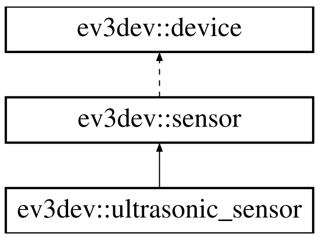
Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

4.69 ev3dev::ultrasonic_sensor Class Reference

Inheritance diagram for ev3dev::ultrasonic_sensor:



Public Member Functions

- **ultrasonic_sensor** (address_type address=INPUT_AUTO)
- float **distance_centimeters** ()
- float **distance_inches** ()
- bool **other_sensor_present** ()

Static Public Attributes

- static const std::string **mode_us_dist_cm** { "US-DIST-CM" }
- static const std::string **mode_us_dist_in** { "US-DIST-IN" }
- static const std::string **mode_us_listen** { "US-LISTEN" }
- static const std::string **mode_us_si_cm** { "US-SI-CM" }
- static const std::string **mode_us_si_in** { "US-SI-IN" }

Additional Inherited Members

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

- /home/panda/Dokumenty/Repos/Ev3Dev/include/ev3dev/ev3dev.h
- /home/panda/Dokumenty/Repos/Ev3Dev/src/ev3dev/ev3dev.cpp

Chapter 5

File Documentation

5.1 /home/panda/Dokumenty/Repos/Ev3Dev/include/action/Action.h File Reference

Contains all Action classes.

```
#include "CommandMotor.h"
#include <memory>
```

Classes

- class ev3::Action

  Base class for all Action controlling classes.
- class ev3::ActionRepeat

  Stores many Actions in a vector and executes them in loop.
- class ev3::ActionDriveDistance

  Implements Robot simple task to drive straight for a given distance.
- class ev3::ActionRotate

  Implements Robot simple task to rotate a given angle, while not driving.
- class ev3::ActionRotateRandDirection

  Implements Robot simple task to rotate a random angle.
- class ev3::ActionStop

  Implements Robot simple task to stop all active motors.
- class ev3::ActionDriveForever

  Implements Robot simple task to drive straight forever.

Typedefs

- typedef std::shared_ptr< Action > ev3::SharedPtrAction

  Type for Action shared_ptr.
- typedef std::vector< SharedPtrAction > ev3::StoredActions

  Type for storing many Actions in one container.
- typedef std::shared_ptr< Command > ev3::SharedPtrCommand

  Type for Command shared_ptr.
- typedef std::vector< SharedPtrCommand > ev3::CommandsVector

  Type for containing associated Command pointers.

5.1.1 Detailed Description

Contains all Action classes.

5.1.2 Typedef Documentation

5.1.2.1 typedef std::vector< SharedPtrAction > ev3::StoredActions

Type for storing many Actions in one container.

See also

ActionRepeat

5.2 /home/panda/Dokumenty/Repos/Ev3Dev/include/action/Behaviour.h File Reference

Contains all Behaviour classes.

```
#include "Action.h"
#include "Utils.h"
#include "Sensor.h"
#include "Event.h"
#include "BehaviourState.h"
#include <unistd.h>
#include <string>
```

Classes

- class ev3::Behaviour

    Base class for all defined behaviours.
- class ev3::BehaviourDriveOnSquare

    Implements complex behaviour of driving on a square-shaped route.
- class ev3::BehaviourExploreRandom

    Implements complex behaviour of exploring the surrounding with random rotation.

Typedefs

- typedef std::shared_ptr< Behaviour > ev3::SharedPtrBehaviour

    Type for Behaviour shared_ptr.
- typedef std::vector< BehaviourState > ev3::BehaviourStates

    Type for storing Behaviour states in one container.
- typedef std::vector< Sensor::SensorType > ev3::Measurements

    Type for storing sensors' desired measurements in one container.

5.2.1 Detailed Description

Contains all Behaviour classes.

5.3 /home/panda/Dokumenty/Repos/Ev3Dev/include/action/BehaviourState.h File Reference

Contains BehaviourState class.

```
#include "Action.h"
#include "Event.h"
```

Classes

- class ev3::BehaviourState

  Encapsulates action and other information in a form of a state.

Typedefs

- typedef std::map< Event::EventType, unsigned int > ev3::ReactionsTransitions

  Type for storing Event-State pairs defining special transitions.

5.3.1 Detailed Description

Contains BehaviourState class.

5.4 /home/panda/Dokumenty/Repos/Ev3Dev/include/communication/Communication.h File Reference

Contains Communication class.

```
#include "Queue.h"
#include "CommUtils.h"
#include <thread>
```

Classes

- class ev3::Communication

  Encapsulates low-level communication and adds logic concerning sending and receiving Message queueing.

5.4.1 Detailed Description

Contains Communication class.

5.5 /home/panda/Dokumenty/Repos/Ev3Dev/include/communication/CommUtils.h File Reference

Contains CommUtils class.

```
#include "Message.h"
#include "CircularBuffer.h"
#include <string>
#include <netinet/in.h>
#include <map>
#include <queue>
```

Classes

- class ev3::CommUtils

  Responsible for low-level communication.
- struct ev3::CommUtils::NetworkNode

  Stores information about a particular node in the network.

5.5.1 Detailed Description

Contains CommUtils class.

5.6 /home/panda/Dokumenty/Repos/Ev3Dev/include/communication/Message.h File Reference

Contains Message class.

```
#include "Utils.h"
#include <vector>
#include <string>
```

Classes

- class ev3::Message

  Stores information passed between physical system units (another robots or master).

5.6.1 Detailed Description

Contains Message class.

5.7 /home/panda/Dokumenty/Repos/Ev3Dev/include/robot/Devices.h File Reference

Contains Devices classes.

```
#include "ev3dev.h"
#include "Motor.h"
#include "Sensor.h"
#include "Utils.h"
```

Classes

- class ev3::Devices

    Singleton class responsible for managing devices connected to the robot.

Variables

- const std::vector< ev3dev::port_type > ev3::INPUTS = {ev3dev::INPUT_1, ev3dev::INPUT_2, ev3dev::IN←PUT_3, ev3dev::INPUT_4}

    Type for storing all available Sensor inputs.
- const std::vector< ev3dev::port_type > ev3::OUTPUTS = {ev3dev::OUTPUT_A, ev3dev::OUTPUT_B, ev3dev::OUTPUT_C, ev3dev::OUTPUT_D}

    Type for storing all available Motor outpus.

5.7.1 Detailed Description

Contains Devices classes.

Index

/home/panda/Dokumenty/Repos/Ev3Dev/include/action/←
Action.h, 99
/home/panda/Dokumenty/Repos/Ev3Dev/include/action/←
Behaviour.h, 100
/home/panda/Dokumenty/Repos/Ev3Dev/include/action/←
BehaviourState.h, 101
/home/panda/Dokumenty/Repos/Ev3Dev/include/communication/←
CommUtils.h, 102
/home/panda/Dokumenty/Repos/Ev3Dev/include/communication/←
Communication.h, 101
/home/panda/Dokumenty/Repos/Ev3Dev/include/communication/←
Message.h, 102
/home/panda/Dokumenty/Repos/Ev3Dev/include/robot/←
Devices.h, 103
_endCondition
ev3::Action, 13

ABORT
ev3::Message, 76
ACTION_FINISHED
ev3::Event, 59
ACTION_INTERR
ev3::Event, 59
ev3::Message, 76
ACTION_OK
ev3::Message, 76
ACK
ev3::Message, 76
AGENT_OVER
ev3::Message, 76
AGENT
ev3::Message, 76
ALL
ev3::LedControl, 69
AMBER
ev3::LedControl, 69
Action
ev3::Action, 11
Action.h
StoredActions, 100
ActionDriveDistance
ev3::ActionDriveDistance, 14
ActionDriveForever
ev3::ActionDriveForever, 16
ActionRepeat
ev3::ActionRepeat, 18
ActionRotate
ev3::ActionRotate, 19
ActionRotateRandDirection
ev3::ActionRotateRandDirection, 21
ActionStop
ev3::ActionStop, 23
ActionType
ev3::Action, 11
addListener
ev3::Devices, 56

BEHAVIOUR_START
ev3::Event, 59
BEHAVIOUR_STOP
ev3::Event, 59
BEHAVIOUR
ev3::Message, 76
BRAKE
ev3::CommandMotorSetStopMode, 46
Behaviour
ev3::Behaviour, 28
BehaviourDriveOnSquare
ev3::BehaviourDriveOnSquare, 31
BehaviourExploreRandom
ev3::BehaviourExploreRandom, 32
BehaviourState
ev3::BehaviourState, 34
BehaviourType
ev3::Behaviour, 28

COAST
ev3::CommandMotorSetStopMode, 46
COLOR
ev3::Sensor, 92
CUSTOM
ev3::Behaviour, 28
checkDevices
ev3::Devices, 56
CommandMotor
ev3::CommandMotor, 40
CommandMotorReset
ev3::CommandMotorReset, 41
CommandMotorRunForever
ev3::CommandMotorRunForever, 42
CommandMotorSetSpeed
ev3::CommandMotorSetSpeed, 43
CommandMotorSetSpeedRegEnabled
ev3::CommandMotorSetSpeedRegEnabled, 44
CommandMotorSetStopMode
ev3::CommandMotorSetStopMode, 46
CommandMotorStop
ev3::CommandMotorStop, 47
CommandSensor
ev3::CommandSensor, 48

createThread
    ev3::Communication, 49
    ev3::Master, 73

DRIVE_DISTANCE
    ev3::Action, 11
DRIVE_FOREVER
    ev3::Action, 11
DRIVE_ON_SQUARE
    ev3::Behaviour, 28
decodeMessage
    ev3::Message, 77
Devices
    ev3::Devices, 55

EMPTY
    ev3::Event, 59
    ev3::Message, 76
EXPLORE_RANDOM
    ev3::Behaviour, 28
empty
    ev3::Message, 77
encodeMessage
    ev3::Message, 77
ev3::Action, 9
    _endCondition, 13
    Action, 11
    ActionType, 11
    DRIVE_DISTANCE, 11
    DRIVE_FOREVER, 11
    getActionPrototype, 12
    getString, 12
    getType, 12
    isExecuted, 12
    isFinished, 12
    NOP, 11
    REPEAT, 11
    ROTATE_RANDOM_DIR, 11
    ROTATE, 11
    STOP, 11
    setCommands, 12
    setEndCondition, 13
ev3::ActionDriveDistance, 13
    ActionDriveDistance, 14
    getActionPrototype, 14
    getDistance, 15
    getString, 15
ev3::ActionDriveForever, 15
    ActionDriveForever, 16
    getActionPrototype, 16
    getString, 17
    isForward, 17
ev3::ActionRepeat, 17
    ActionRepeat, 18
    getString, 18
ev3::ActionRotate, 18
    ActionRotate, 19
    getActionPrototype, 20
    getRotation, 20
    getString, 20
ev3::ActionRotateRandDirection, 20
    ActionRotateRandDirection, 21
    execute, 21
    getActionPrototype, 22
    getString, 22
ev3::ActionStop, 22
    ActionStop, 23
    getActionPrototype, 23
    getString, 23
ev3::Agent, 24
    getCommId, 24
    getId, 24
    processMessage, 25
    setBehaviour, 25
    setCommId, 25
    setId, 25
    setMeasurement, 25
    updateLastMessage, 26
ev3::Behaviour, 26
    Behaviour, 28
    BehaviourType, 28
    CUSTOM, 28
    DRIVE_ON_SQUARE, 28
    EXPLORE_RANDOM, 28
    getPrototype, 28
    getString, 29
    react, 29
    setMeasurements, 29
    setReactionStates, 29
    setStates, 29
    setStopState, 30
ev3::BehaviourDriveOnSquare, 30
    BehaviourDriveOnSquare, 31
    getPrototype, 31
    getString, 31
ev3::BehaviourExploreRandom, 32
    BehaviourExploreRandom, 32
    getPrototype, 32
    getString, 33
ev3::BehaviourState, 33
    BehaviourState, 34
    getAction, 34
    getReaction, 35
    isStopState, 35
    process, 35
    setNextState, 35
    setReactions, 35
ev3::CircularBuffer< T >, 36
ev3::ColorUtils, 37
ev3::CommUtils, 50
    preparePassiveSocket, 50
    receiveMessage, 50
    receiveMessageDelay, 51
    sendMessage, 51
ev3::CommUtils::NetworkNode, 83
ev3::Command, 38
    getString, 39

ev3::CommandMotor, 39
  CommandMotor, 40
  getMotor, 40
ev3::CommandMotorReset, 41
  CommandMotorReset, 41
ev3::CommandMotorRunForever, 42
  CommandMotorRunForever, 42
ev3::CommandMotorSetSpeed, 43
  CommandMotorSetSpeed, 43
ev3::CommandMotorSetSpeedRegEnabled, 44
  CommandMotorSetSpeedRegEnabled, 44
ev3::CommandMotorSetStopMode, 45
  BRAKE, 46
  COAST, 46
  CommandMotorSetStopMode, 46
  HOLD, 46
  StopMode, 46
ev3::CommandMotorStop, 46
  CommandMotorStop, 47
ev3::CommandSensor, 47
  CommandSensor, 48
  getSensor, 48
ev3::Communication, 48
  createThread, 49
  run, 49
ev3::Devices, 54
  addListener, 56
  checkDevices, 56
  Devices, 55
  getInstance, 56
  getMotor, 56
  getSensor, 57
  operator=, 57
  removeListener, 57
  setProximitySensor, 57
  setSafetyTouchSensor, 57
ev3::Event, 58
  ACTION_FINISHED, 59
  ACTION_INTERR, 59
  BEHAVIOUR_START, 59
  BEHAVIOUR_STOP, 59
  EMPTY, 59
  Event, 59
  EventType, 59
  getStringType, 59
  getType, 59
  OBSTACLE_DETECTED, 59
  PROXIMITY_ALERT, 59
  SENSOR_WATCH, 59
ev3::EventAction, 60
  EventAction, 60
  getActionType, 61
ev3::EventQueue, 61
ev3::EventSensorWatch, 62
  EventSensorWatch, 62
  getType, 63
  getValue, 63
ev3::LedControl, 68
  ALL, 69
  AMBER, 69
  flash, 69
  flashColor, 69
  GREEN_ALL, 69
  GREEN_L, 69
  GREEN_R, 69
  GREEN, 69
  LedColors, 69
  LedType, 69
  off, 70
  on, 70
  onExclusive, 70
  RED_ALL, 69
  RED_L, 69
  RED_R, 69
  RED, 69
  setColor, 70
  YELLOW, 69
ev3::Logger, 72
ev3::Master, 73
  createThread, 73
  run, 73
  send, 74
ev3::Message, 75
  ABORT, 76
  ACTION_INTERR, 76
  ACTION_OK, 76
  ACK, 76
  AGENT_OVER, 76
  AGENT, 76
  BEHAVIOUR, 76
  decodeMessage, 77
  EMPTY, 76
  empty, 77
  encodeMessage, 77
  getMessageId, 77
  getParameters, 78
  getReceiverId, 78
  getSenderId, 78
  getString, 78
  getType, 78
  MASTER_OVER, 76
  MASTER, 76
  MEASURE, 76
  Message, 76
  MessageType, 76
  NOT, 76
  PAUSE, 76
  PING, 76
  PONG, 76
  RESUME, 76
  SENSOR_VALUE, 76
  START, 76
  setMessageId, 78
  setParameters, 79
  setReceiverId, 79
  setSenderId, 79

setType, 79
ev3::Motor, 80
getMotor, 80
Motor, 80
ev3::Queue< T >, 84
ev3::Robot, 85
ev3::RobotModelA, 86
ev3::RobotState, 86
ev3::RobotStateActive, 87
ev3::RobotStateIdle, 88
ev3::RobotStatePanic, 88
ev3::RobotStatePaused, 89
ev3::RobotStateWorking, 89
ev3::Sensor, 92
COLOR, 92
GYRO, 92
getDecimals, 93
getNumValues, 93
getSensor, 93
getType, 93
getValue, 93
getValueF, 93
INFRARED, 92
LIGHT, 92
prepareMessage, 94
SOUND, 92
Sensor, 92
SensorType, 92
TOUCH, 92
ULTRASONIC, 92
ev3::SignalHandler, 95
ev3dev::button, 36
ev3dev::color_sensor, 37
ev3dev::dc_motor, 52
ev3dev::device, 53
ev3dev::gyro_sensor, 63
ev3dev::i2c_sensor, 64
ev3dev::infrared_sensor, 65
ev3dev::large_motor, 65
ev3dev::lcd, 66
ev3dev::led, 66
ev3dev::lego_port, 71
ev3dev::light_sensor, 71
ev3dev::medium_motor, 74
ev3dev::motor, 81
ev3dev::power_supply, 83
ev3dev::remote_control, 84
ev3dev::sensor, 90
ev3dev::servo_motor, 94
ev3dev::sound, 95
ev3dev::sound_sensor, 96
ev3dev::touch_sensor, 96
ev3dev::ultrasonic_sensor, 97
Event
ev3::Event, 59
EventAction
ev3::EventAction, 60
EventSensorWatch
ev3::EventSensorWatch, 62
EventType
ev3::Event, 59
execute
ev3::ActionRotateRandDirection, 21

flash
ev3::LedControl, 69
flashColor
ev3::LedControl, 69

GREEN_ALL
ev3::LedControl, 69
GREEN_L
ev3::LedControl, 69
GREEN_R
ev3::LedControl, 69
GREEN
ev3::LedControl, 69
GYRO
ev3::Sensor, 92
getAction
ev3::BehaviourState, 34
getActionPrototype
ev3::Action, 12
ev3::ActionDriveDistance, 14
ev3::ActionDriveForever, 16
ev3::ActionRotate, 20
ev3::ActionRotateRandDirection, 22
ev3::ActionStop, 23
getActionType
ev3::EventAction, 61
getCommId
ev3::Agent, 24
getDecimals
ev3::Sensor, 93
getDistance
ev3::ActionDriveDistance, 15
getId
ev3::Agent, 24
getInstance
ev3::Devices, 56
getMessageId
ev3::Message, 77
getMotor
ev3::CommandMotor, 40
ev3::Devices, 56
ev3::Motor, 80
getNumValues
ev3::Sensor, 93
getParameters
ev3::Message, 78
getPrototype
ev3::Behaviour, 28
ev3::BehaviourDriveOnSquare, 31
ev3::BehaviourExploreRandom, 32
getReaction
ev3::BehaviourState, 35
getReceiverId

ev3::Message, 78
getRotation
ev3::ActionRotate, 20
getSenderId
ev3::Message, 78
getSensor
ev3::CommandSensor, 48
ev3::Devices, 57
ev3::Sensor, 93
getString
ev3::Action, 12
ev3::ActionDriveDistance, 15
ev3::ActionDriveForever, 17
ev3::ActionRepeat, 18
ev3::ActionRotate, 20
ev3::ActionRotateRandDirection, 22
ev3::ActionStop, 23
ev3::Behaviour, 29
ev3::BehaviourDriveOnSquare, 31
ev3::BehaviourExploreRandom, 33
ev3::Command, 39
ev3::Message, 78
getStringType
ev3::Event, 59
getType
ev3::Action, 12
ev3::Event, 59
ev3::EventSensorWatch, 63
ev3::Message, 78
ev3::Sensor, 93
getValue
ev3::EventSensorWatch, 63
ev3::Sensor, 93
getValueF
ev3::Sensor, 93

HOLD
ev3::CommandMotorSetStopMode, 46

INFRARED
ev3::Sensor, 92
isExecuted
ev3::Action, 12
isFinished
ev3::Action, 12
isForward
ev3::ActionDriveForever, 17
isStopState
ev3::BehaviourState, 35

LIGHT
ev3::Sensor, 92
LedColors
ev3::LedControl, 69
LedType
ev3::LedControl, 69

MASTER_OVER
ev3::Message, 76
MASTER
ev3::Message, 76
MEASURE
ev3::Message, 76
Message
ev3::Message, 76
MessageType
ev3::Message, 76
Motor
ev3::Motor, 80

NOP
ev3::Action, 11
NOT
ev3::Message, 76

OBSTACLE_DETECTED
ev3::Event, 59
off
ev3::LedControl, 70
on
ev3::LedControl, 70
onExclusive
ev3::LedControl, 70
operator=
ev3::Devices, 57

PAUSE
ev3::Message, 76
PING
ev3::Message, 76
PONG
ev3::Message, 76
PROXIMITY_ALERT
ev3::Event, 59
prepareMessage
ev3::Sensor, 94
preparePassiveSocket
ev3::CommUtils, 50
process
ev3::BehaviourState, 35
processMessage
ev3::Agent, 25

RED_ALL
ev3::LedControl, 69
RED_L
ev3::LedControl, 69
RED_R
ev3::LedControl, 69
REPEAT
ev3::Action, 11
RESUME
ev3::Message, 76
RED
ev3::LedControl, 69
ROTATE_RANDOM_DIR
ev3::Action, 11
ROTATE

ev3::Action, 11
react
ev3::Behaviour, 29
receiveMessage
ev3::CommUtils, 50
receiveMessageDelay
ev3::CommUtils, 51
removeListener
ev3::Devices, 57
run
ev3::Communication, 49
ev3::Master, 73

SENSOR_VALUE
ev3::Message, 76
SENSOR_WATCH
ev3::Event, 59
SOUND
ev3::Sensor, 92
START
ev3::Message, 76
STOP
ev3::Action, 11
send
ev3::Master, 74
sendMessage
ev3::CommUtils, 51
Sensor
ev3::Sensor, 92
SensorType
ev3::Sensor, 92
setBehaviour
ev3::Agent, 25
setColor
ev3::LedControl, 70
setCommId
ev3::Agent, 25
setCommands
ev3::Action, 12
setEndCondition
ev3::Action, 13
setId
ev3::Agent, 25
setMeasurement
ev3::Agent, 25
setMeasurements
ev3::Behaviour, 29
setMessageId
ev3::Message, 78
setNextState
ev3::BehaviourState, 35
setParameters
ev3::Message, 79
setProximitySensor
ev3::Devices, 57
setReactionStates
ev3::Behaviour, 29
setReactions
ev3::BehaviourState, 35
setReceiverId
ev3::Message, 79
setSafetyTouchSensor
ev3::Devices, 57
setSenderId
ev3::Message, 79
setStates
ev3::Behaviour, 29
setStopState
ev3::Behaviour, 30
setType
ev3::Message, 79
StopMode
ev3::CommandMotorSetStopMode, 46
StoredActions
Action.h, 100

TOUCH
ev3::Sensor, 92

ULTRASONIC
ev3::Sensor, 92
updateLastMessage
ev3::Agent, 26

YELLOW
ev3::LedControl, 69