Generic Robot

Instructions on Creating a Project (what to do on kickoff day)

Decoupling HW from code

Allows Unit Testing (w/ mocking)

Allows us to leave core code alone, and just work on year-specific code

Training: go here, run this Junit test. See how … Change something, see what happens

Taurus library

Controllers (Taurus)

Joystick axes

Buttons / Axis Buttons / POV Buttons

Separation of concerns – HW declared outside Controller, passed into constructor

Message Queues & Readers

XboxController / ThrustmasterController / ButtonBoarder

Unit Test & Mockito

Drive/Operator Controls (competition specific)

Interfaces

Implementations

MessageQueues

GenericQueues & Unit Test

FlatBuffers

MessageQueues & Unit Test

MessageQueueManager

Logging

nestedBuffers <https://groups.google.com/forum/#!topic/flatbuffers/aP74NvqwNpw>

forceVectorAlignment <https://github.com/google/flatbuffers/commit/07da3fc216c62b18eb13a8bcb9afa95d7c325418>

RobotInit()

Read config file

For each channel listed, check if this is the first time this channel is mentioned

If it is the first time, create a queue

newQueue = Class.forName(“frc.taurus.messages.MessageQueue<classListedInConfigFile>”).getInstance();

Queues use their class name to autogenerate the getRootAs function name. Error messages if not found

Where are these queues stored??? In a container in MessageQueueManager. A Map. A Sorted Map. A HashMap?

Map<Class, MessageQueue<Type>).> queueMap;

If channel is to be logged to file, register the GenericQueue<ByteBuffer> with the Logger, along with the filename to write to

Modules will write messages. Modules will know what Type of messages they want to send

Modules ask MesssageQueueManager where the queue for this type is

MessageQueue<JoystickStatus> localQueue = MessageQueueManager.getQueue<JoystickStatus.class>;

Watchers will also generate a localReader = localQueue.makeReader();

Loggers will get readers to GenericQueue<ByteBuffer>. An array of struct [queue, filename] needs to be stored

RobotPeriodic()

Modules will write messages.

localQueue.writeMessage(JoystickStatus status);

Modules will read messages

JoystickGoal goal = localQueue.readLastMessage();

Loggers will log all messages to file

For each (reader : logReaderList)

While reader is not empty

Msg = Reader.getNextMessage();

Pack into packet

Write(packet)