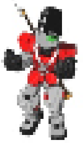
Introduction to C++ on the 2009 FRC Control System

Pat Fairbank & Leigh Pauls – November 8th, 2008

Patrick Fairbank

- ▶ 9 years of *FIRST* experience
- ▶ Mentor for Team 296, 2004 – 2006
  - ▶ 2006 World Champions
- ▶ Mentor for Team 1503, 2007 – Present
  - ▶ 2 regional finalists
- ▶ University of Waterloo undergrad student
  - ▶ Mechatronics Engineering, Class of 2011



Introduction to C++

Overview
Major Differences
WindRiver
WPILib

Basics
IterativeRobot
printf

Driver Station I/O
Joystick
Analog Input
Digital I/O

Robot I/O
Speed Controller
Servo
Relay
Solenoid
Analog Input
Digital I/O

Convenience Classes
Compressor
Encoder
Gyro
Timer
Others

Camera

Questions?

Major Differences From 2008

- ►C++ classes are used to represent everything
- ►More power, memory
  - ►Floating-point calculations are now okay
  - ►32-bit calculations are cheaper
- ►On-board clock makes real-time timing easy

Introduction to C++

Overview
- Major Differences
- WindRiver
- WPILib

Basics
- IterativeRobot
- printf

Driver Station I/O
- Joystick
- Analog Input
- Digital I/O

Robot I/O
- Speed Controller
- Servo
- Relay
- Solenoid
- Analog Input
- Digital I/O

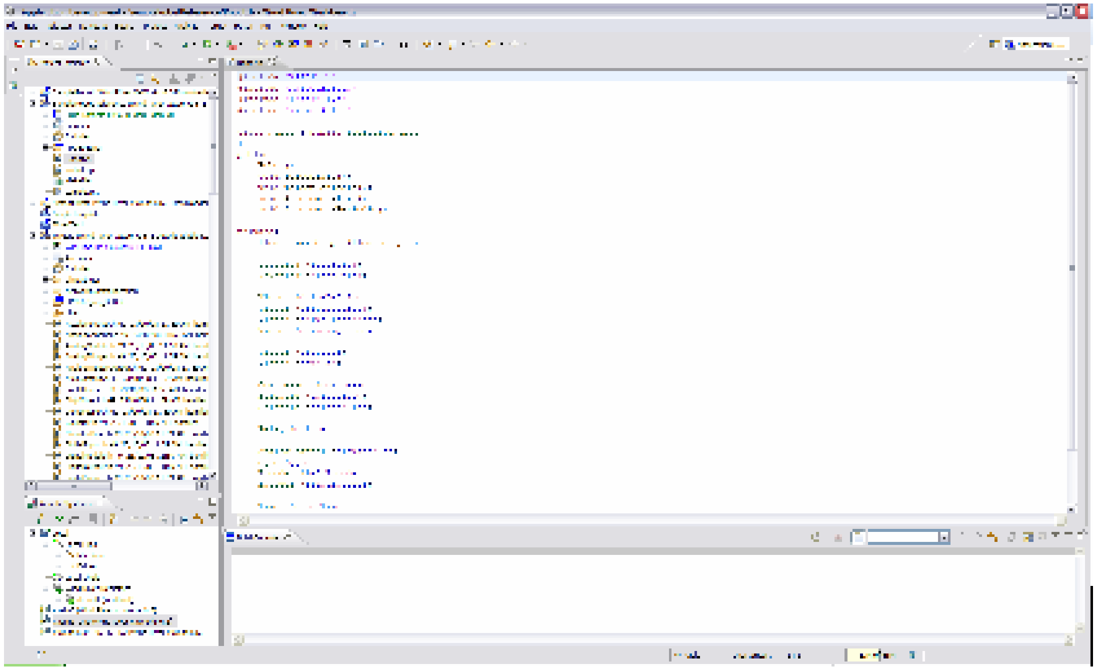
Convenience Classes
- Compressor
- Encoder
- Gyro
- Timer
- Others

Camera

Questions?

Wind River

- ► Embedded systems company
- ► Workbench 3.0 based on Eclipse
- ► Compiler used is GCC
- ► VxWorks RTOS runs on cRIO



WPILib

- The library we use for C++ programming on the new controller
- Developed mainly by WPI
- Collection of classes representing everything to do with the control system
- Each class has .h and .cpp files
- Easiest source of documentation for a class is the .h file

Introduction to C++

Overview
Major Differences
WindRiver
WPILib

Basics
IterativeRobot
printf

Driver Station I/O
Joystick
Analog Input
Digital I/O

Robot I/O
Speed Controller
Servo
Relay
Solenoid
Analog Input
Digital I/O

Convenience Classes
Compressor
Encoder
Gyro
Timer
Others

Camera

Questions?

IterativeRobot

- A base class for robot programs
- A robot program is made by creating a new class which inherits from IterativeRobot
- Customization is achieved in the constructor and by implementing 1 or more of 6 important virtual functions

IterativeRobot Example

```
class MyRobotClass : public IterativeRobot
{
public:
  MyRobotClass();
  void DisabledInit();
  void DisabledPeriodic();
  void TeleopInit();
  void TeleopPeriodic();
  void AutonomousInit();
  void AutonomousPeriodic();

private:
  // Member variables
  Joystick *driveJoystick;
  Victor *leftDrive;
  Victor *rightDrive;
}
```

Constructor

- Called once when the robot is first turned on or reset
- Guaranteed to be called before any other member functions in the class

```
MyRobotClass::MyRobotClass()
{
  // Create member variables
  driveJoystick = new Joystick(1);
  leftDrive = new Victor(1);
  rightDrive = new Victor(2);
}
```

DisabledInit

- Called automatically once at the beginning of disabled mode

```
void MyRobotClass::DisabledInit()
{
  // Code to be run once when the robot is first turned on at
  // competition, or whenever the robot is first disabled
}
```

Introduction to C++

Overview
Major Differences
WindRiver
WPILib

Basics
IterativeRobot
printf

Driver Station I/O
Joystick
Analog Input
Digital I/O

Robot I/O
Speed Controller
Servo
Relay
Solenoid
Analog Input
Digital I/O

Convenience Classes
Compressor
Encoder
Gyro
Timer
Others

Camera

Questions?

DisabledPeriodic

- Called on a 200 Hz cycle (every 5 ms) while the robot is disabled

```
void MyRobotClass::DisabledPeriodic()
{
  GetWatchdog()->Feed();
  // Code to be run periodically while the robot is disabled
  // E.g. Choose autonomous modes from the driver station
}
```

TeleopInit

- Called automatically once at the beginning of teleoperated mode

```
void MyRobotClass::TeleopInit()
{
  // Code to be run once when the robot is first enabled for driver
  // control
  // E.g. Initialize outputs to zero
}
```

TeleopPeriodic

- Called on a 200 Hz cycle (every 5 ms) while the robot is enabled for driver control

```
void MyRobotClass::TeleopPeriodic()
{
  GetWatchdog()->Feed();
  // Code to be run periodically while the robot is enabled
  // E.g. Mapping of joystick inputs to motor outputs
}
```

Introduction to C++

Overview
Major Differences
WindRiver
WPILib

Basics
IterativeRobot
printf

Driver Station I/O
Joystick
Analog Input
Digital I/O

Robot I/O
Speed Controller
Servo
Relay
Solenoid
Analog Input
Digital I/O

Convenience Classes
Compressor
Encoder
Gyro
Timer
Others

Camera

Questions?

AutonomousInit

- Called automatically once at the beginning of autonomous mode

```
void MyRobotClass::AutonomousInit()
{
  // Code to be run once when the robot is first enabled for
  // autonomous
  // E.g. Initialize outputs to zero, reset autonomous counters
}
```

AutonomousPeriodic

- ►Called on a 200 Hz cycle (every 5 ms) while the robot is enabled for autonomous mode

```
void MyRobotClass::TeleopPeriodic()
{
  GetWatchdog()->Feed();
  // Code to be run periodically while the robot is in autonomous
  // E.g. Autonomous control of motors using timing and sensors
}
```

Introduction to C++

Overview
Major Differences
WindRiver
WPILib

Basics
IterativeRobot
printf

Driver Station I/O
Joystick
Analog Input
Digital I/O

Robot I/O
Speed Controller
Servo
Relay
Solenoid
Analog Input
Digital I/O

Convenience Classes
Compressor
Encoder
Gyro
Timer
Others

Camera

Questions?

printf

- ► Used for displaying text and variables to the cRIO’s internal terminal
- ► Syntax is the same as printf on the old system

Examples:

```
printf(“This is a line of output\n”);
// Displays “This is a line of output.”

int someInteger = 5;
printf(“Some integer = %d\n”, someInteger);
// Displays “Some integer = 5”

float someFloat = 2.71828;
printf(“Some integer = %d, some float = %f\n”, someInteger, someFloat);
// Displays “Some integer = 5, someFloat = 2.71828”
```

Joystick

- Represents a device plugged into one of the 4 USB ports on the Driver Station

Constructor:

```
Joystick(UINT32 port)
```

Important member functions:

```
float GetX(), float GetY(), bool GetTrigger(), bool GetRawButton(UINT32 button)
```

Example:

```
Joystick *leftJoystick;
leftJoystick = new Joystick(1);        // USB Port 1

float output = leftJoystick->GetY();
if (leftJoystick->GetTrigger()) {
  // Do something conditional on the joystick trigger being pressed
}
```

DS Analog Input

- 4 analog inputs (potentiometers, etc.)
- Accessed through the single instance of the DriverStation class

Important member functions:

```
float GetAnalogIn(UINT32 channel)
```

Example:

```
DriverStation *driverStation;
driverStation = DriverStation::GetInstance();

float dsAnalog2 = driverStation->GetAnalogIn(2);    // Analog input 2
```

DS Digital Input/Output

- ►8 digital inputs (buttons), 8 digital outputs (LEDs)
- ►Accessed through the single instance of the DriverStation class

Important member functions:

void GetDigitalIn(UINT32 channel), void SetDigitalOut(UINT32 channel)

Example:

```
DriverStation *driverStation;
driverStation = DriverStation::GetInstance();

bool dsDigIn4 = driverStation->GetDigitalIn(4);// Digital input 4
driverStation->SetDigitalOut(3, true);          // Turn on digital output 3
```

Speed Controller

- ▶ Represents a Victor or Jaguar plugged into one of the 10 PWM outputs on the DSC
- ▶ Output value ranges from -1.0 to 1.0

Constructor:

Victor(UINT32 channel), Jaguar(UINT32 channel)

Important member functions:

void Set(float value)

Example:

```
Victor *rightDrive;
Jaguar *arm;
rightDrive = new Victor(1);          // PWM 1
arm = new Jaguar(2);                 // PWM 2

rightDrive->Set(1.0);                // Full speed forward
arm->Set(0.0);                       // Stopped
```

Introduction to C++

Overview
Major Differences
WindRiver
WPILib

Basics
IterativeRobot
printf

Driver Station I/O
Joystick
Analog Input
Digital I/O

Robot I/O
Speed Controller
Servo
Relay
Solenoid
Analog Input
Digital I/O

Convenience Classes
Compressor
Encoder
Gyro
Timer
Others

Camera

Questions?

Servo

- Represents a servo plugged into one of the 10 PWM outputs on the DSC
- Output value can range from -1.0 to 1.0, or can be in degrees (typically 0.0 to 270.0)

Constructor:

Servo(UINT32 channel)

Important member functions:

void Set(float value), void SetAngle(float angle)

Example:

```
Servo *cameraServo;
cameraServo = new Servo(9);         // PWM 9

cameraServo->Set(0.0);              // Middle of servo range
cameraServo->SetAngle(35.0);        // 35 degrees from lower bound position
```

Introduction to C++

Overview
Major Differences
WindRiver
WPILib

Basics
IterativeRobot
printf

Driver Station I/O
Joystick
Analog Input
Digital I/O

Robot I/O
Speed Controller
Servo
Relay
Solenoid
Analog Input
Digital I/O

Convenience Classes
Compressor
Encoder
Gyro
Timer
Others

Camera

Questions?

Relay

- ► Represents a Spike plugged into one of the 8 relay outputs on the DSC
- ► Output value is *kOff*, *kForward* or *kReverse*

Constructor:

```
Relay(UINT32 channel)
```

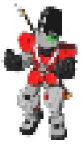
Important member functions:

```
void Set(Value value)
```

Example:

```
Relay *rollerMotor;
rollerMotor = new Relay(4);            // Relay 4

Relay->Set(Relay::kForward);
```



Solenoid

- Represents a pneumatic solenoid valve plugged into one of the 8 solenoid outputs on the Solenoid Breakout
- Double solenoid valve requires two
- Output value is true or false

Constructor:

```
Solenoid(UINT32 channel)
```

Important member functions:

```
void Set(bool on)
```

Example:

```
Solenoid *gripperCylinder;
gripperCylinder = new Solenoid(4);                // Solenoid 4

gripperCylinder->Set(true);
```

Introduction to C++

Overview
Major Differences
WindRiver
WPILib

Basics
IterativeRobot
printf

Driver Station I/O
Joystick
Analog Input
Digital I/O

Robot I/O
Speed Controller
Servo
Relay
Solenoid
Analog Input
Digital I/O

Convenience Classes
Compressor
Encoder
Gyro
Timer
Others

Camera

Questions?

Analog Input

- Represents an analog sensor plugged into one of the 8 inputs on the Analog Breakout
- Input value is a 12-bit integer or a floating point voltage

Constructor:

```
AnalogChannel(UINT32 channel)
```

Important member functions:

```
INT16 GetValue(), float GetVoltage()
```

Example:

```
AnalogChannel *armPot;
armPot = new AnalogChannel(6);              // Analog input 6

INT16 armValue = armPot->GetValue();        // Ranges from -2048 to 2047
float armVoltage = armPot->GetVoltage();    // Ranges from -10V to 10V
```

Introduction to C++

Overview
Major Differences
WindRiver
WPILib

Basics
IterativeRobot
printf

Driver Station I/O
Joystick
Analog Input
Digital I/O

Robot I/O
Speed Controller
Servo
Relay
Solenoid
Analog Input
Digital I/O

Convenience Classes
Compressor
Encoder
Gyro
Timer
Others

Camera

Questions?

Digital Input/Output

- 14 GPIO ports on the DSC can be configured for input or output

Constructor:

```
DigitalInput(UINT32 channel), DigitalOutput(UINT32 channel)
```

Important member functions:

```
UINT32 Get()                    // DigitalInput
void Set(UINT32 value)          // DigitalOutput
```

Example:

```
DigitalInput *limitSwitch;
DigitalOutput *robotLed;
limitSwitch = new DigitalInput(2);          // Digital I/O 2
robotLed = new DigitalOutput(3);            // Digital I/O 3

UINT32 limitValue = limitSwitch->Get();     // Returns either 0 or 1
robotLed->Set(1);                           // Set to either 0 or 1
```

Overview
Major Differences
WindRiver
WPILib

Basics
IterativeRobot
printf

Driver Station I/O
Joystick
Analog Input
Digital I/O

Robot I/O
Speed Controller
Servo
Relay
Solenoid
Analog Input
Digital I/O

Convenience Classes
Compressor
Encoder
Gyro
Timer
Others

Camera

Questions?

Compressor

- ► Represents an air compressor plugged into a relay module with a pressure switch plugged into a digital input
- ► Uses a DigitalInput and a Relay in the background
- ► Set and forget

Constructor:

```
Compressor(UINT32 pressureSwitchChannel, UINT32 compressorRelayChannel)
```

Important member functions:

```
void Start()
```

Example:

```
Compressor *compressor;
compressor = new Compressor(2, 1);          // Pressure switch on DI/O 2, relay 1
Compressor->Start();
```

Introduction to C++

Overview
Major Differences
WindRiver
WPILib

Basics
IterativeRobot
printf

Driver Station I/O
Joystick
Analog Input
Digital I/O

Robot I/O
Speed Controller
Servo
Relay
Solenoid
Analog Input
Digital I/O

Convenience Classes
Compressor
Encoder
Gyro
Timer
Others

Camera

Questions?

Encoder

- ▶ Represents a quadrature encoder plugged into two digital inputs on the DSC
- ▶ Counts rising and falling edge of both channels, so pulse count is multiplied by 4

Constructor:

```
Encoder(UINT32 aChannel, UINT32 bChannel, bool reverseDirection = false)
```

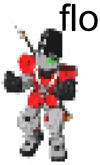
Important member functions:

```
void Start(), INT32 Get(), void Reset(), UINT32 GetPeriod()
```

Example:

```
Encoder *leftEncoder;
leftEncoder = new Encoder(11, 12);              // A channel is DI/O 11, B is 12
leftEncoder->Start();

INT32 distance = leftEncoder->Get();
float speed = leftEncoder->GetPeriod() * 1000000;          // In pulses per second
```



Gyro

- Represents a gyro (yaw rate sensor)
- Must be plugged into Analog Input 1 because it has a hardware accumulator

Constructor:

```
Gyro(UINT32 channel)
```

Important member functions:

```
float GetAngle(), void Reset(), void SetSensitivity(float voltsPerDegreePerSecond)
```

Example:

```
Gyro *gyro;
gyro = new Gyro(1);                // Analog input 1
gyro->SetSensitivity(0.0122);      // Sensitivity for a particular model of gyro

float robotAngle = gyro->GetAngle();
```

Timer

- ► Measures passage of time in microseconds
- ► Useful for time-based autonomous modes

Constructor:

```
Timer()
```

Important member functions:

```
void Start(), void Reset(), UINT32 Get()
```

Example:

```
Timer *autonTimer;
autonTimer = new Timer();
autonTimer->Start();

if (autonTimer->Get() > 1000000)
{
  autonTimer->Reset();
  // Do something after a delay of 1 second
}
```

Introduction to C++

Overview
Major Differences
WindRiver
WPILib

Basics
IterativeRobot
printf

Driver Station I/O
Joystick
Analog Input
Digital I/O

Robot I/O
Speed Controller
Servo
Relay
Solenoid
Analog Input
Digital I/O

Convenience Classes
Compressor
Encoder
Gyro
Timer
Others

Camera

Questions?

Other Useful Classes

- Accelerometer
- GearTooth
- HiTechnicCompass
- Ultrasonic

Introduction to C++

Overview
Major Differences
WindRiver
WPILib

Basics
IterativeRobot
printf

Driver Station I/O
Joystick
Analog Input
Digital I/O

Robot I/O
Speed Controller
Servo
Relay
Solenoid
Analog Input
Digital I/O

Convenience Classes
Compressor
Encoder
Gyro
Timer
Others

Camera

Questions?

Camera

- Useful for identifying blobs of colour
- Used to track targets
- Works with groups of ‘Particles’
- Can be used to move to a coloured target with PID control
- Can identify the size of coloured targets

Introduction to C++

Overview
Major Differences
WindRiver
WPILib

Basics
IterativeRobot
printf

Driver Station I/O
Joystick
Analog Input
Digital I/O

Robot I/O
Speed Controller
Servo
Relay
Solenoid
Analog Input
Digital I/O

Convenience Classes
Compressor
Encoder
Gyro
Timer
Others

Camera

Questions?

Particles

- The data from the camera
- A single ‘blob’ of colour in an image
- Describes the image of:
  - A light
  - A colored object
  - Whatever the Game Design Committee wants you to find

Particle Members

- Center of Mass x / y
  - The position of the blob on the camera
  - Weighted average of the area that the blob covers
- Surface Area
- Rectangular Bounds
- Percent of Image
- Particle Quality

Defining a Color

- Define a particle in Hue/Saturation/Luminosity (HSL)
  - Hue
    - The pigment of the colour
    - How ‘red’ or how ‘blue’ a colour is
  - Saturation
    - How intensely the Hue is applied to the colour
    - Controls how vibrant or dull a colour is
  - Luminosity
    - How bright the colour is
    - Controls the difference between white/pale and black/dark colours

Overview
Major Differences
WindRiver
WPILib

Basics
IterativeRobot
printf

Driver Station I/O
Joystick
Analog Input
Digital I/O

Robot I/O
Speed Controller
Servo
Relay
Solenoid
Analog Input
Digital I/O

Convenience Classes
Compressor
Encoder
Gyro
Timer
Others

Camera

Questions?

Initialize the Camera

- Start Sampling images from the Camera

```
// start the camera
if (StartCameraTask(10, 0, k160x120, ROT_0) == -1) {
    dprintf( LOG_ERROR,"Failed to spawn camera task;Error
    code %i",GetErrorText(GetLastError()));
}
```

- Pick a colour to start looking for

```
// values for tracking a target -may need tweaking in your
    environment
TrackingThreshold data = GetTrackingData(GREEN,
    PASSIVE_LIGHT);
```

Introduction to C++

Overview
Major Differences
WindRiver
WPILib

Basics
IterativeRobot
printf

Driver Station I/O
Joystick
Analog Input
Digital I/O

Robot I/O
Speed Controller
Servo
Relay
Solenoid
Analog Input
Digital I/O

Convenience Classes
Compressor
Encoder
Gyro
Timer
Others

Camera

Questions?

Custom Colour Range

- ► Pick your own colour with a range of HSL values

```
TrackingThreshold tdata;   // image data for tracking

//HSL values for an active green light
tdata.hue.minValue = 67;
tdata.hue.maxValue = 114;

tdata.saturation.minValue = 161;
tdata.saturation.maxValue = 255;

tdata.luminance.minValue = 24;
tdata.luminance.maxValue = 101;
```

Use a Particle

- Decide How large the Particle can be

```
#define MIN_PARTICLE_TO_IMAGE_PERCENT 0.25      // target is too small
#define MAX_PARTICLE_TO_IMAGE_PERCENT 10.0      // target is too close
```

- Get a Particle, and decide if it is large enough to be the light

```
// look for the colour
if (FindColor(IMAQ_HSL, &tdata.hue, &tdata.saturation,
      &tdata.luminance, &par)
      && par.particleToImagePercent <
      MAX_PARTICLE_TO_IMAGE_PERCENT
      && par.particleToImagePercent >
      MIN_PARTICLE_TO_IMAGE_PERCENT)
{
```

- Use the information in the particle for something

```
      double xin = par.center_mass_x_normalized;
      double yin = par.center_mass_y_normalized;

      printf("found: x: %f y: %f\n", xin, yin);
}
```

Introduction to C++

Overview
- **Major Differences**
- **WindRiver**
- **WPILib**

Basics
- **IterativeRobot**
- **printf**

Driver Station I/O
- **Joystick**
- **Analog Input**
- **Digital I/O**

Robot I/O
- **Speed Controller**
- **Servo**
- **Relay**
- **Solenoid**
- **Analog Input**
- **Digital I/O**

Convenience Classes
- **Compressor**
- **Encoder**
- **Gyro**
- **Timer**
- **Others**

Camera

Questions?

Questions?

► This presentation and other resources will be posted to the *FIRST* Beta Test Public Forum:

http://forums.usfirst.org/forumdisplay.php?f=743

► Feel free to send any C++ questions to patfair@gmail.com