

# Introduction to Qu@sar

This document is intended as a starting point for the [Qu@sar](#) documentation. This document should be read, along with the Primer document, for a complete introduction to [Qu@sar](#). A basic understanding of Python (<http://www.python.org>) is assumed. A tutorial can be found at <http://www.python.org/doc/current/tut/tut.html>.

Qu@sar is a developer's toolkit for interacting with Q330<sup>1</sup> digitizers and PB14 packet balers from Quanterra. Written in Python<sup>2</sup>, Qu@sar works properly on any platform that Python is available for, including Windows, MacOS, and most un\*x-like OSs. Effort has been made to allow Qu@sar to run properly under Jython<sup>3</sup>, therefore interacting with Java applications, or even writing a Q330 enabled Java application should be fairly easy.

For lower-level uses, Qu@sar completely implements the QDP protocol, allowing packets to be created, sent to, received from, and examined. From these building blocks, an unlimited number of Q330 aware applications can be easily written and maintained.

For higher-level needs, Qu@sar implements several high level functions that are commonly needed when dealing with Q330s and Balers, for example registration and cloning. These higher level functions are provided in the spirit of code centralization.

## Goals

- Easy to use  
Qu@sar is being designed with the goal of allowing the programmer to get the job done with as little distraction from the task at hand as possible. This means having an easy to understand, well documented API.
- Portable  
Qu@sar needs to run on as many platforms as possible, and allow code to work as intended on every supported platform with no modification required.
- Scalable  
Qu@sar must not only support existing Q330s, but also support future devices that may use the QDP protocol, and future versions of the Q330. New command packets may be required, but the underlying code should be flexible enough to permit this.

## Basics

Included is a direct implementation of the QDP protocol. There is a

---

<sup>1</sup> Information on the Q330 can be found at <http://www.q330.com>

<sup>2</sup> Python can be found at <http://www.python.org>

<sup>3</sup> Jython can be found at <http://www.jython.org>

QDPPacket class, which is then extended by each of the classes in the “Commands” package.

The QDPPacket class handles all of the conversions to and from the “string of bytes” that is required for transmission on the wire, as well as all byte order issues. Also handled inside the QDPPacket base class is the conversion to string, required for “printing” a packet.

The concept of “things that can talk QDP” is implemented in the QDPDevice class. Methods such as sendCommand() and connection management exist here. This is where the code that actually puts QDPPacket instances (and instances of derived classes) on the wire. This class is extended in the Q330 and Baler classes, which add higher level, device specific functionality, such as register, configuration management, and data retrieval.

There are also several convenience classes, for dealing with configuration profiles, and token memory.

## (I)FAQs

Q: Why Python?

A: Python was chosen because it's very easy to learn, yet powerful enough to do anything. The goal was to pick a language that would not become an obstacle in the end user's quest to write scripts. Furthermore, Python has quite an impressive following, including RedHat and Google. For a list of companies using Python, see <http://www.python.org/community/users.html>

Q: What about platform independence?

A: Qu@sar will run on any platform that Python is available for. This includes, but is not limited to: Windows, MacOS, just about any Unix (or un\*x-like) OS, and even the Java Virtual Machine (using Jython)

Q: What about byte order issues?

A: Byte order issues will all be handled transparently to the end user. The same scripts will run on any platform.

Q: Will Qu@sar do \_\_\_\_\_?

A: Our goal is to make Qu@sar a committee designed product. If there is enough demand for a feature, and the feature is reasonable to implement, it will be included.