3. Problem Shooting

When you get stuck working with TinyOS, two good resources are the TinyOS FAQ (http://www.tinyos.net/faq.html) and discussion archives (https://mail.millennium.berkeley.edu/pipermail/). Most likely any problem you will encounter, someone has seen it before. So before posting a question to a discussion forum, make sure it has not been answered before or ask each other through the WebBoard that was set up for this course.

Here are some answers to possible problems you might encounter.

(1) Telos does not support some of the example applications that the manual refers to, as they were originally written for MICA motes. E.g. SimpleCmd and SenseLightToLog.
(2) In lesson 2, the Telos motes use DemoSensorC.nc from /platform/msp430, which is wired to the InternalTempC. The light sensor on the device is in HamamatsuC.nc in /platform/telos. There is also a "total solar radiation" (TSR) and a "photosynthetically active radiation" (PAR) sensor. For more information on these sensors, check out the datasheet on the Mote*iv* or Crossbow website.
(3) The color of the standard LED implementation does not match the actual LEDs on the Telos. In the example applications that we provided, there is a new component LedsTelosC.nc with interface LedsTelos.nc.
(4) For timer variables, always use *unique("Timer")*, since this is also used in some system functions (such as GenericComm.nc).
(5) In case you are wondering, TOS_Msg for Telos is in /tos/platform/telos/AM.h. When you look at the output of the Listen tool in lesson 6, the fields do not seem to correspond with what is described in the tutorial. The reason is that the message format is slightly different for Telos. What you see is related to the TOS_Msg in /tos/platform/telos/AM.h (not in /tos/types/AM.h).
(6) The java classes generated by "mig" and the AM message type are not always compatible. When an 8-bit variable is used in an AM message, it is possible that it is padded by 8 zeros. In fact, within an AM message, each 16-bit variable apparently must start at an offset that is a multiple of 16. However, "mig" does not take this padding into account. It is therefore advisable to first list all 16-bit variables in your message definition and only then the 8-bit ones.
(7) If you create a message that is too long, the send routine will get messed up and no packets are sent (the AM data size is 28 bytes, which is 14 uint16_t).
(8) Be careful when sending data over both the UART and the radio, as they could block each other.
(9) The Telos motes use the CC2420 radio. Its features can be controlled, see also /tos/lib/CC2420Radio. Examples that use this are in /contrib/ucd/apps/. You can connect CC2420Control to CC2420RadioC and use that interface. Also check out the CC2420 datasheet for a list of the control registers. http://www.chipcon.com/index.cfm?kat_id=2&subkat_id=12&dok_id=115.