Handling Video File Date/Time

What's Going On with Date/Time

GoPro's `file.lastModified()` (and sometimes EXIF timestamps) often **store the time in "local time"**, but **label it as if it were UTC**.

There is some logic to this, as if the camera is accurately set to the “local time” it can be saved in a way that you can get that “local time” back no matter what time zone the camera was in. One just needs to read the time, and format it as though it came from a UTC timezone.

This means:

The timestamp is a naive "local time" value, but it's interpreted by Java (or other languages) as UTC , so when calling `.format()` — if calibrated to the local timezone, it would lead to the time appearing 7 or 8 hours too early, but if the formatter is set to UTC timezone it will yield the accurate time

So my original code worked because the Date formatter (that converts a Long to a Date, adjusts for the timezone (does a -7 for PDT) so me doing a +7 before that canceled that out, and we are left with the time being formatted as if it was UTC time or the recording local time

So, to display the local time that the video was taken, in Java (can translate to other languages) just format the Date/Time: Long value to UTC time

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss a");
sdf.setTimeZone(TimeZone.getTimeZone("UTC")); // <-- Treat timestamp as-is
System.out.println("Local time shown as UTC: " + sdf.format(new Date(f.lastModified())));

where new Date(f.lastModified()) is the value of the Date/Time:[ ] parameter *1000

The Gate/Black magic Date/Time:[ ] are saved the same way, and yield the right time (at least at Shortline), need to make sure they are encoded right no matter the timezone