ICS 314 Team project
Spring 2015
Calendaring Assignment based on RFC 5545

Your team assignment is a straightforward implementation using common software engineering tools. This is not difficult, but requires some attention to detail. Although you can find calendar libraries to do some or all of the assignment, I want your team to implement this yourselves.

We will adopt the agile idea of "sprints". You will have three check-ins to make sure things are on track and to demo what you've accomplished.

All of you have experience with Eclipse (http://www.eclipse.org/), so use Eclipse for your software development. You will be hosting your project online, and using revision control (git, or subversion, …).

A significant task, outlined below, is to create .ics "event files", as described in RFC 5545 (https://tools.ietf.org/html/rfc5545). Of course, you won't be implementing everything that's in RFC 5545! *In fact don't know of any commercial implementation of everything in RFC 5545*, so you'll be doing a small piece.

Since you are implementing the Internet standard for calendar events, you can use an existing calendaring system like Outlook, Google Calendar, or Mac OS Calendar to read files (hint: that's a good way to test the output of your system). You can also use those calendars to create .ics event files that you can view with a text editor like vim (hint: a great way to reverse engineer event descriptions).

Every team member on each team shall **turn in the same pdf document**, i.e., one document per team.

Check-in 1
Deliverable due: Friday 13 March
Points: 4

You need to host your team project on github (https://github.com/) or google code (https://code.google.com/hosting/createProject)github

You will **not** be developing a fancy user interface, but don't forget about input validation. Your interface can be command-line or a simple web-like interface where users can choose dates, times, and other input values.

Deliverable: Turn in a single pdf document with the following:

- URL of the where you code is hosted
- The number of lines of code developed to-date, and the programming language used.
- A screen shot showing the .ics file generated for this event: *Studying for finals, Hamilton Library, 6-10pm HST 8 May 2015, high priority, public*.
- A screen shot showing a calendar (google, Mac Calendar, Outlook, …) after reading in the event file you just created
- Any questions or comments about the calendaring project so far.

You are developing a stand-alone application to create .ics event files. You must implement the following:

- Version (section 3.7.4 of RFC 5545)
- Classification (3.8.1.3). Note this is a way of users designating events as public (default), private, or confidential.
- Location (3.8.1.7)
- Priority (3.8.1.9)
- Summary (3.8.1.12)
- DTSTART (3.8.2.4)
- DTEND (3.8.2.2)
- Time zone identifier (3.8.3.1, and whatever other sections you need to be able to specify time zones)

Note that Section 4 of the RFC has examples of .ics files, and Section 5 is Recommended Practices.

Check-in 2
Deliverable due: Friday 3 April (Note that 23-27 March is spring break)
Points: 5

You must use Test Driven Development (TDD) or assert ()'s for the remaining of the assignment. If you are programming in Java use JUnit (there is a similar tool for C++).

For Check-in 2 you shall implement the following: The input is a list of .ics filenames (pathnames, if necessary) that describe events all on the same date and in the same time zone. The output is .ics files describing free times between the inputted events. The Summary field of the output .ics files shall be "Free Time".

Deliverable: Turn in a single pdf document with the following:

- The number of lines of code developed to-date, including the increase or decrease in LOC from previous check-in
- A description of number of TDD tests or asserts used. Include the ratio of test cases or asserts to total LOC
- A screen shot of a calendar (google, Mac Calendar, Outlook) displaying the input files
- A screen shot of a calendar (google, Mac Calendar, Outlook, …) dislpaying the input and the output (free time) files
- Any questions or comments about the calendaring project so far.

Check-in 3
Deliverable due: Friday 17 April
Points: 4

Take at least two lists of .ics file names, describing events on the same day and time zone, and find *common* free times, generating .ics files for common free times. The idea is that one input list is for Person A's events that day, and the other input list is for Person B's events that day. The output is possible meeting times for Person A and B. The Summary field of the common free times shall be "POSSIBLE MEETING TIME"

Deliverable: Turn in a single pdf document with the following

- The number of lines of code developed to-date, including the increase or decrease in LOC from previous check-in
- A description of number of TDD tests or asserts used. Include the ratio of test cases or asserts to total LOC
- A screen shot of a calendar (google, Mac Calendar, Outlook) displaying the two lists of input files (Person A and Person B's events for a particular day)
- A screen shot of a calendar (google, Mac Calendar, Outlook, …) showing the input and the output (possible meeting) files
- Any questions or comments about the calendaring project so far.

Bonus point: take in to account priority times – find the best common meeting time that could include low priority busy times.

Finale
Deliverable due: Wednesday 6 May
Points: 5

- The URL for a 5 minute *narrated* video demonstration of your final project. *Your demo shall not be longer than five minutes.* You can use jing.com, screencast-o-matic.com, …., or if you have a Mac, QuickTime, to capture your screen to video. You need to demonstrate the functionality described below.
- A one-page pdf document reporting the number of LOC, and convince me that your system is correct. That is, how did you test your system and what were the results? Hint: I know you will have used TDD or asserts, but that isn't enough.
- Submit a URL for your code repository. I need to be able to see the code, and a log of your commits,.