Copilot

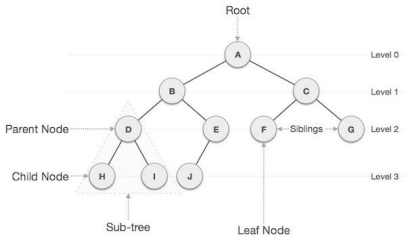
Embedding turn-by-turn trail directions on a watch

Why do this?

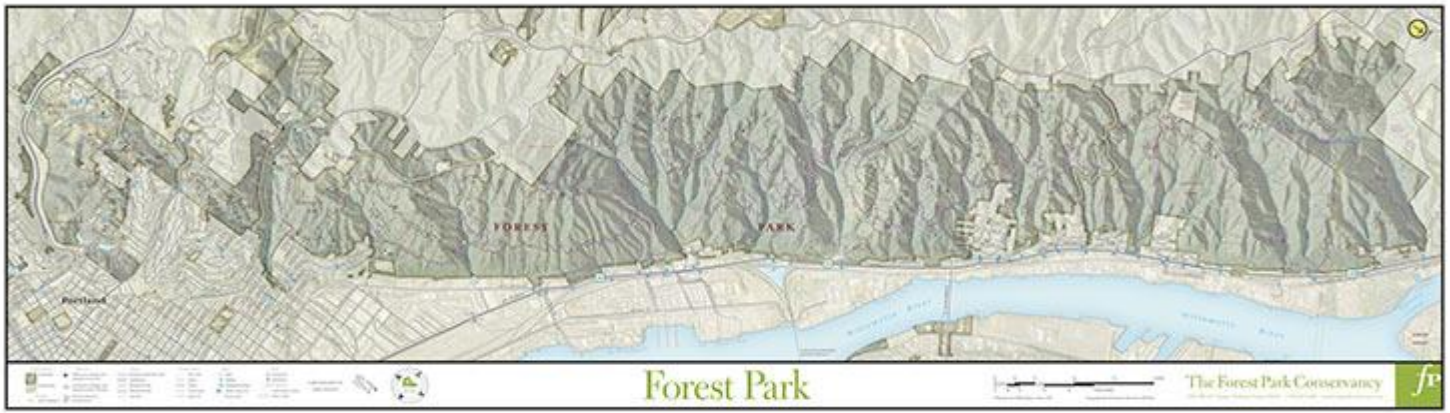
- Trail runners are a feisty bunch… they expect
  - You'll always know the route
  - Route directions will always be provided
  - You'll never need to pause a conversation to look at a map
  - The mileage you provide will 100% always match their GPS watch
- Writing software for tiny platforms is fun

PDX Trail Runners

Wouldn’t it be cool if...



- All of Forest Park could be represented in a single data structure?
- That you could use along with a specified route to print out turn-by-turn direction labels?
- And upload onto a wearable of some kind to provide instant navigation capabilities during a run?



But which wearable to target?

The Pebble smart watch seemed like the best fit (as of mid-2015)

Pebble Specs (I owned a Pebble Time, C SDK)

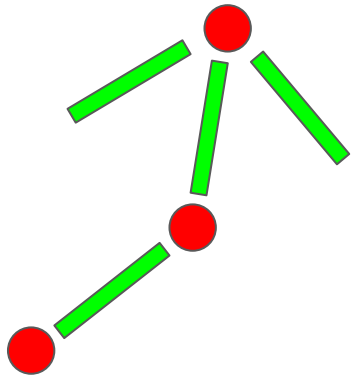
Feature	Classic, Steel	Time, Time Steel	Time Round	Pebble 2	Time 2
Platform	Aplite	Basalt	Chalk	Diorite	Emery
CPU	Cortex-M3 64 MHz	Cortex-M4 100 MHz			Cortex-M7 144 MHz
Max. resource size	96k	256k			512k
Max. app size (code + heap)	24k	64k			128k
Display shape	Rectangle		Round	Rectangle	
Display resolution	144 x 168		180 x 180	144 x 168	200 x 228
Display PPI	175	182		175	202
Supported colors	2	64		2	64
Heart Rate Monitor	No	Yes *(with smartstrap)*		Yes *(except SE model)*	Yes
Microphone	No	Yes			
Sensors	Accelerometer, Compass			Accelerometer	Accelerometer, Compass
Buttons	4				
Charging port	Power only	Smart accessory port		Smart accessory port *(data only)*	Smart accessory port

But first, some prerequisite work...

- Figure out how to structure the (80+ miles of) trails in the park
- Decide how to structure specific routes
- Provide a way to create new routes
- Provide a means to visualize both the routes and the entire park for easy reference

How to structure the trail data?

- For every path (portion of trail between 2 junctions) in the park:
  - Measure the distance of the path
  - Assign a unique integer ID for the path
- For every junction in the park:
  - Take a compass bearing for each outward path
  - Assign a name to each outward path
  - Associate the integer ID of each outward path
  - Assign a unique integer ID for the junction
- A route is simply a junction ID followed by a sequence of path IDs



How to structure the trail data?

```
337     def buildDatabase(self):
338         self.clearAll()
339
340         # 3/28/2015
341         self.addName('Wildwood Access', 'WLD')
342         self.addName('Newton Access', 'NWT')
343         self.addName('Wildwood', 'WLD')
344         self.addName('Firelane 10', 'F10')
345         self.addName('Newton Rd', 'NWT')
346         self.addName('BPA Rd', 'BPA')
347         self.addName('Firelane 15', 'F15')
348         self.addName('Firelane 12', 'F12')
349         self.addName('Firelane 13', 'F13')
```

```
420
421         # 3/28/2015
422         self.addPath(1,  0.10)
423         self.addPath(2,  0.33)
424         self.addPath(3,  0.15)
425         self.addPath(4,  0.16)
426         self.addPath(5,  0.39)
427         self.addPath(6,  0.15)
428         self.addPath(7,  0.21)
429         self.addPath(8,  1.15)
430         self.addPath(9,  0.39)
431         self.addPath(10, 1.28)
432         self.addPath(11, 0.46)
```

```
# 3/28/2015
self.addJunction(1, [[1, 46, 'Wildwood Access'], [7, 326, 'Newton Rd'], [6, 140, 'Firelane 10'], [21, 246, 'Newton Rd']])
self.addJunction(2, [[2, 16, 'Wildwood'], [3, 168, 'Wildwood'], [1, 204, 'Newton Access']])
self.addJunction(3, [[5, 298, 'Wildwood'], [4, 208, 'Newton Access'], [2, 120, 'Wildwood']])
self.addJunction(4, [[10, 2, 'Newton Rd'], [8, 228, 'Wildwood'], [9, 180, 'Newton Rd'], [5, 150, 'Wildwood']])
self.addJunction(5, [[4, 30, 'Wildwood Access'], [9, 340, 'Newton Rd'], [7, 160, 'Newton Rd']])
self.addJunction(6, [[14, 210, 'BPA Rd'], [12, 20, 'BPA Rd'], [8, 46, 'Wildwood']])
self.addJunction(7, [[11, 232, 'BPA Rd'], [13, 350, 'Wildwood'], [14, 46, 'BPA Rd']])
self.addJunction(8, [[20, 210, 'Firelane 15'], [15, 290, 'Wildwood'], [16, 2, 'Firelane 15'], [13, 82, 'Wildwood']])
self.addJunction(9, [[18, 290, 'Firelane 12'], [17, 110, 'Firelane 12'], [16, 156, 'Firelane 15']])
self.addJunction(10, [[19, 338, 'BPA Rd'], [12, 150, 'BPA Rd'], [17, 250, 'Firelane 12']])
self.addJunction(11, [[21, 324, 'Newton Rd']])
self.addJunction(12, [[10, 122, 'Newton Rd'], [22, 312, 'Newton Rd'], [23, 162, 'BPA Rd']])
self.addJunction(13, [[22, 208, 'Newton Rd']])
self.addJunction(14, [[24, 312, 'Firelane 13'], [23, 346, 'BPA Rd'], [19, 138, 'BPA Rd']])
```

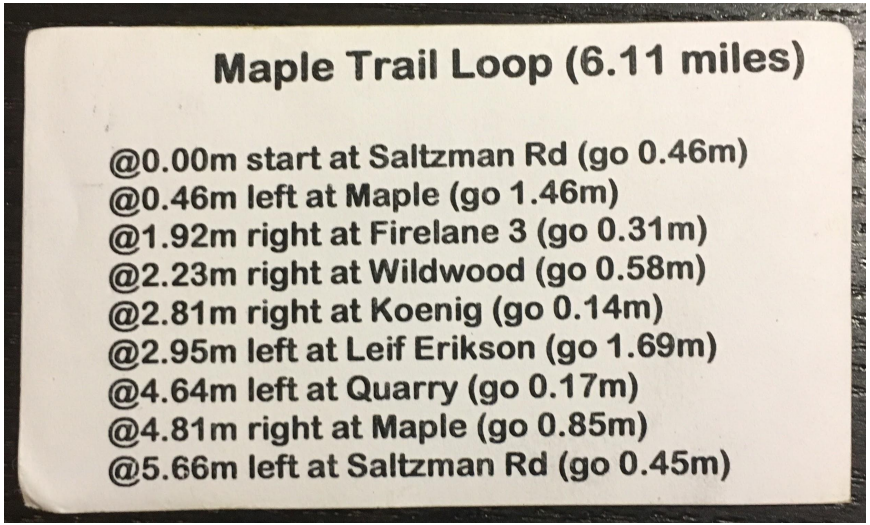
Applications

Now that a data model exists for the park, we can write applications to

- Print out turn-by-turn directions for a route
- Construct mathematical curves to visualize the data model
- Build arbitrary routes for different runs
- Give turn-by-turn directions for a route as it’s being traversed

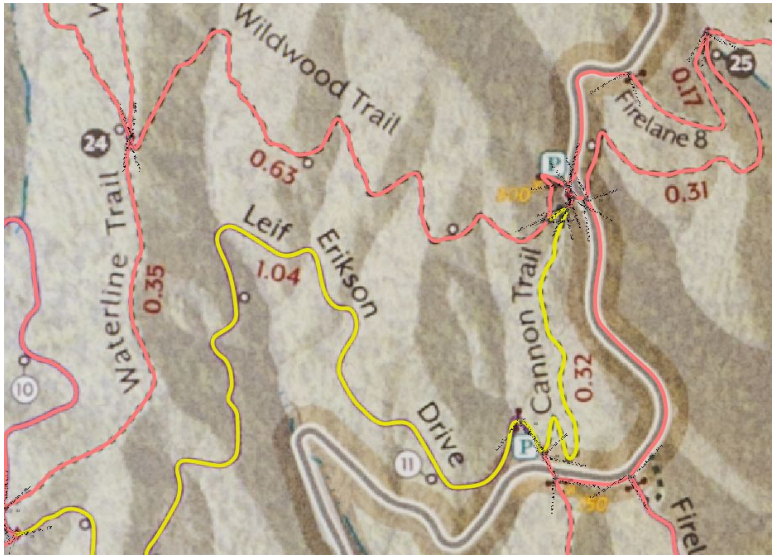
Applications - Route Stickers

- Generated with a Python script
- Walks the route, keeping track of mileage and turns along the way



Applications - Route Maker

A C++ / Qt application to visualize trails/routes



Applications - Pebble App

- A C application to provide turn-by-turn directions
- Quiet “buzz” when nearing a junction
- Requires no GPS! (Rate*Time = Distance)
- Gives est. mileage remaining
- Tracks PR’s
- The entire app, including database, fits in < 64K RAM
- Pause/Resume a run
- Stores run data in EEPROM

What's next?

- Pebble Time 2 was due to be released shortly after I designed the first version of the application
  - It was going to be amazing!
  - Bigger Screen!
  - More RAM!
- But then they went out of business (assets acquired by FitBit)
- Fast forward 2 years… FitBit creates the Versa, which looks suspiciously similar to the never released Pebble Time 2!
  - Even more amazing!
  - Even bigger screen!
- But it's super buggy and rushed to market :(
- Not sure what's next, but after doing all this work, I know the trails so well that... I don't really need a watch anymore