Not the intended topic

• I intended this lecture to begin a series of lectures on the subject of testing and debugging.

• What I found in your assignments suggested this example instead.

• Which is about **not** testing and debugging (or rather, not needing to)
Geolocation API

- The HTML 5 Geolocation API is at dev.w3.org/geo/api/spec-source.html
Simple, easy, wrong.

- var location = navigator.location()
- var longitude = location.longitude ...
- if (-48 <= latitude && latitude <= -32 && 164 <= longitude && longitude <= 180 ) yep_it_is_in_New_Zealand()
What’s the method?

• In Smalltalk I’d do something like
  location := GeographicLocation current.
  No abbreviations or meaningless prefixes

• In JavaScript, is it location, geolocation,
  geographiclocation, or what? How are
  word boundaries marked? is geo+location
  a boundary or not?
Find-documentation or IntelliSense (feh!)

• If “navigator” has a fixed meaning, the IDE can know what its interface is.

• It’s “geolocation”

• The IDE cannot complain about location because this is legal:

navigator.location = navigator.geolocation
Problem 1

• The sample code I wrote is **legal** JavaScript

• Before that assignment, navigator.location is a **legal** expression answering “undefined”, not an error!

• But navigator.location.longitude is an error
Let’s fix that

- var location = navigator.geolocation
- var longitude = location.longitude ...
- This is a **legal** declaration. The value happens to be “undefined”, is all.
- location.longitude = 170

is **legal**, after which location.longitude=170
What just happened?

- `navigator.geolocation.longitude = 170` assigns 170 to a slot of a function.
- Remember, all functions are objects. Assigning to a property of a function when you meant to assign to a property of an(other kind of) object is legal.
Why is it a function?

- The user might have switched off location support; you have to ask if it’s OK to use the current location.
- In a mobile device, running the GPS unit takes electrical power; turning it off saves the battery.
- In a moving device, the location can change.
Sigh. Let’s fix it.

• var location = navigator.geolocation()
• var longitude = location.longitude ... 
• Whoops.
• Double whoops. It’s not a function.
It’s an object with function fields

• method getCurrentPosition()  
  — where am I now?

• method watchPosition()  
  — start tracking the position

• method clearWatch()  
  — stop tracking the position
Let’s try again again.

• `var location = navigator.geolocation.getCurrentPosition()`
• `var longitude = location.longitude`
• `Error: TypeError: DOM Exception 17`
• `WTH?`
In the console

- `> navigator.geolocation.getCurrentPosition`
- `< function getCurrentPosition() {
  [native code]
} }

- This shows a function of no arguments, doesn’t it?
No

• `navigator.geolocation.getCurrentPosition.length`
• `is 1`
• `Length is so obviously the word for arity or argument count, isn’t it?`
• `Gosh this is hard!`
What the API says

• callback PositionCallback =
  
  void (Position position)

• callback PositionErrorCallback =
  
  void (PositionError positionError)

• void getCurrentPosition(PositionCallback success, optional PositionErrorCallback error, optional PositionOptions options)
It’s complex, but clear

- `getCurrentPosition` **does not get the current position.** It starts an activity which will at some future time determine the current position and pass it to the success callback or fail and pass the reason to the failure callback.

- This is asynchronous programming and it is a serious pain in the posterior.
What we have to do

• var location

• navigator.geolocation.getCurrentPosition(
  function (position) { location = position})

• // much later

• if (location !== undefined) ...
We’re still not home

• location.longitude is legal but “undefined”

• interface Position {

  readonly attribute Coordinates coords;

  readonly attribute DomTimeStamp timestamp;
}

• Why? Because position can change.
Heading for home

• interface Coordinates {
  readonly attribute double latitude;
  readonly attribute double longitude;
  readonly attribute double? altitude;
  readonly attribute double accuracy;
  readonly attribute double? altitudeAccuracy;
  readonly attribute double? heading;
  readonly attribute double? speed; }

So where are we?

• if (location !== undefined) {
    longitude = location.coords.longitude
    latitude = location.coords.latitude
...
• Such a lot to get wrong!
Type = documentation

• If we have the types before us,
  • The IDE can prompt us
  • We can prevent errors before running the code.
• So why not let us describe our own interfaces the same way?
Type ≠ enough doc!

• What are latitude and longitude?

• The API spec says in English that they “are geographic coordinates specified in decimal degrees”

• “Decimal” here is seriously misleading.

• And sigh, geographic location is not that simple...