6.170 – Software Studio
Objects

Adam Chlipala, Mark Day, Daniel Jackson
September 15, 2014
Possibly of Interest:

Next Monday, a seminar talk on mathematical formalization of JavaScript program behavior, using computer theorem proving.

Not part of 6.170, but you're all welcome to attend!

Alan Schmitt, “JSCert, a two-pronged approach to JavaScript formalization.”

September 22, 2 PM, 32-D463

https://calendar.csail.mit.edu/seminar_series/6767
Object-Oriented Programming!
Who could say “no” to that?
But what is it?

JavaScript doesn't have these!

Dynamic Dispatch?
method calls on same variable
   call different functions at runtime

Encapsulation?
some state inside an object is
   private

Records?
packaging together related
   fields and methods, referred to
   by name

Classes?
“just like in Java/C++”

Subtyping?
like with Java interfaces

Inheritance?
like with Java extends

JavaScript sort of has this.
What are Objects in JavaScript?

Short Answer:
Dictionaries [like in Python]
+ Prototypes [probably new to most of you]

No counterpart to uses of static typing in Java.

Other elements of OO from Java are built on top of functions!
What are Objects in General?

Boxes of *encapsulated mutable fields* that accept *method calls* from outside of themselves.

Code can be written *polymorphically*, so that it depends only on the methods available in objects, not on their private fields.

We already saw most of this with functions. Objects add *dictionaries* and *prototype inheritance*. 
Dictionaries

```
foo  42
bar  "hi"
baz

qux
quux ...
...  ...
```
When looking up a field that is not found, try the same lookup in the prototype of the object, and so on for the prototype of that object, etc.

When we start out with first-class functions, this pair of features is all we need to recreate usual object-oriented programming!
Pragmatics of Objects in JavaScript

Back to JavaScript Live!
http://people.csail.mit.edu/dnj/teaching/6170/javascript-live/

(We'll see a number of features designed to make JavaScript smell more like Java, but which may actually be more trouble than they're worth [e.g., new keyword].)
The Expression Problem [putting OO in perspective]

Two dimensions of extensibility, a key feature for evolvability of software:

Adding new types of objects: strong point of object orientation
Adding new methods: strong point of functional programming & recursive data structures

<table>
<thead>
<tr>
<th></th>
<th>add</th>
<th>multiply</th>
<th>subtract</th>
<th>divide</th>
<th>.......</th>
</tr>
</thead>
<tbody>
<tr>
<td>naturals</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>integers</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>reals</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.......</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Demo: A DOM Animation Library

using a relatively minimalist take on JavaScript OO.

Code in:

http://github.com/6170-fa14/6.170-lectures
Miscellaneous

• Project 1.1 due tomorrow; 1.2 a week later.
• Thursday's recitation: CSS, the visual styling system of the web
• *Next Monday*: second *in-class quiz!* On this week's material about objects and GUIs.