Objective-J

Objective-J is a programming language developed as part of the Cappuccino web development framework. Its syntax is nearly identical to the Objective-C syntax and it shares with JavaScript the same relationship that Objective-C has with the C programming language: that of being a strict, but small, superset; adding traditional inheritance and Smalltalk/Objective-C style dynamic dispatch. Pure JavaScript, being a prototype-based language, already has a notion of object orientation and inheritance, but Objective-J adds the use of class-based programming to JavaScript.

Programs written in Objective-J need to be preprocessed before being run by a web browser's JavaScript virtual machine. This step can occur in the web browser at runtime or by a compiler which translates Objective-J programs into pure JavaScript code. The Objective-J compiler is written in JavaScript; consequently, deploying Objective-J programs does not require a web browser plug-in. Objective-J can be compiled and run on Node.js.

Objective-J

Paradigm	Multi-paradigm: reflective, object-oriented, functional, imperative, scripting
Developer	Cappuccino Core Developers and community.
First appeared	2008
Typing discipline	dynamic, weak, duck
License	LGPL
Website	cappuccino-project.org (http://www.cappuccino-project.org/)
Influenced by	
Objective-C, JavaScript	

Contents

Applications

The first widely known use of Objective-J was in the Cappuccino-based web application 280 Slides, which was developed by 280 North itself. Even though Objective-J can be used (and has been designed) independently from the Cappuccino framework, Objective-J has primarily been invented to support web development in Cappuccino.

Applications designed using the Cappuccino Framework[1]

- RW Elephant (http://www.rwelephant.com/)
- Mockingbird (http://gomockingbird.com/)
- GithubIssues (http://githubissues.herokuapp.com/)

- Enstore (https://web.archive.org/web/20110226082843/http://www.enstore.com/) (until October 2013, they rewrote it using Ember [2])

Syntax

Objective-J is a superset of JavaScript, which means that any valid JavaScript code is also valid Objective-J code.

The following example shows the definition and implementation in Objective-J of a class named Address; this class extends the root object CPObject, which plays a role similar to the Objective-C's NSObject. This example differs from traditional Objective-C in that the root object reflects the underlying Cappuccino framework as opposed to Cocoa, Objective-J does not use pointers and, as such, type definitions do not contain asterisk characters. Instance variables are always defined in the @implementation.

```
@implementation Address : CPObject
{
  CPString name;
  CPString city;
}

- (id)initWithName:(CPString)aName city:(CPString)aCity
{
  self = [super init];

  name = aName;
  city = aCity;

  return self;
}

- (void)setName:(CPString)aName
{
  name = aName;
}

- (CPString)name
{
  return name;
}

+ (id)newAddressWithName:(CPString)aName city:(CPString)aCity
{
  return [[self alloc] initWithName:aName city:aCity];
}

@end
```

As with Objective-C, class method definitions and instance method definitions start with '+' (plus) and '-' (dash), respectively.

Memory management

Objective-C uses ARC (Automatic Reference Counting) for deallocating unused objects. In Objective-J, objects are automatically deallocated by JavaScript's Garbage Collector.

See also

- Cappuccino (application development framework)

References

1. "Demos in Cappuccino" (https://www.webcitation.org/6J8WaykuI?url=http://www.cappuccino-project.org/learn/demos.html). *Demos in Cappuccino*. Archived from the original (http://cappuccino.org/learn/demos/) on 25 August 2013. Retrieved 26 February 2011.
2. http://blog.acclivitynyc.com/post/64981755172/dumped-cappuccino-and-switched-to-ember

External links

- Official website (http://cappuccino-project.org/)
- "Learning Objective-J" (http://cappuccino-project.org/learn/tutorials). Cappuccino Web Framework.