A Glimpse on Javascript

Name: Xiang Gao
GWID: G44616546

Introduction

I choose Javascript to write my project which is a simple web-based cost calculator. As I thought this project mainly operate on front-end, thus I choose the most popular front-end language, Javascript, to do this project.

Javascript provides many features like dynamic typing, prototype-based and functional. It uses syntax influenced by the language C. JavaScript copies many names and naming conventions from Java.

Names

There's no length restriction in variable/function names, however, there are some restrictions varies on different javascript interpreters.

Names in Javascript are case-sensitive.

Variables

Javascript is an object-based language, which means everything in Javascript is an object. So variable can have it's own properties.

Javascript is a dynamic typing language, a variable is not required to have its type declared.

Types

Javascript use dynamic, weakly typed typing system.

In Javascript, types are associated with values, not with variables.

JavaScript supports various ways to test the type of an object, including duck typing.

Duck typing is a style of dynamic typing in which an object's methods

and properties determine the valid semantics, rather than its inheritance from a particular class or implementation of a specific interface.

Expressions

Javascript has the following types of expressions:

- Arithmetic: evaluates to a number, for example 3.14159.
- String: evaluates to a character string, for example, "Fred" or "234". (Generally uses string operators.)
- Logical: evaluates to true or false. (Often involves logical operators.)
- Object: evaluates to an object.

Control Structures

Javascript is very much like C, the control statements are very similar, including `if`, `while`, `do ... while`, `for` and `switch`.

The usage are almost the same as in C and other C-like languages.

Subprograms

This is one of the Javascript's most competitive features. Subprograms are one of the most powerful building blocks of modular programming. It helps to break down a complex task into small pieces. Javascript has a good support of modular design. One reason that Javascript is popular today is that there are tons of useful modules(subprograms) you can easily imported to your program.

Abstract Data Types

Javascript has many built-in abstract data types, and user can easily built their own abstract data types from the Prototype type. JavaScript possess all the familiar object-oriented programming tenets, by its ability to define new object types and extend them through its prototype property. Prototypes serve as templates for an entire class of abstract data types. Therefore prototype properties are shared by all instantiated objects of the class. While this is a powerful and expressive type of inheritance model, what's missing is the customary class inheritance support that is

familiar to programmers of conventional object-oriented languages.

Encapsulation

Javascript use module pattern and closure to achieve encapsulation. The use of anonymous functions (closure) can encapsulate functions in defined objects. For example:

```
function Person(name) {
        var _name = name;
        return {
        setName: function(n) { _name = n; },
        getName: function() { return _name; }
        }
}
```

Object Oriented Support

The two core features in Javascript to achieve Object Oriented support is Object and Prototype.

An object is a member of the type Object. It is an unordered collection of properties each of which contains a primitive value, object, or function. A function stored in a property of an object is called a method.

A prototype is an object used to implement structure, state, and behavior inheritance. When a constructor creates an object, that object implicitly references the constructor's associated prototype for the purpose of resolving property references. The constructor's associated prototype can be referenced by the program expression constructor.prototype, and properties added to an object's prototype are shared, through inheritance, by all objects sharing the prototype.

There are also some built-in object types.

Exception Handling and Event Handling

Javascript has builtin control structure to handling exceptions. The try/catch/finally statement of Javascript lets you capture exceptions and deal with it.

Concurrency

Javascript doesn't support parallel processing it self. But there are many extension libraries to provide easy parallel processing. These extensions are heavily depends on the environment/interpreter. Currently, most modern interpreters can run Javascript parallel implicitly.

Conclusion

Javascript (often shortened to JS) is a lightweight, interpreted, object-oriented language with first-class functions, most known as the scripting language for Web pages, but used in many non-browser environments as well such as backbone.js, node.js or Apache CouchDB.

References

http://www.codeproject.com/Articles/5608/Writing-Object-Oriented-JavaScript-Part-1

http://en.wikipedia.org/wiki/JavaScript

https://developer.mozilla.org/en-US/docs/JavaScript