1.1 Pattern

"theme of recurring events or objects… it can be a template or model which can be used to generate things" (<http://en.wikipedia.org/wiki/Pattern>).

• Design patterns - *Elements of Reusable Object-Oriented Software.*

• Coding patterns - JavaScript-specific patterns and good practices related to the unique features of the language, such as the various uses of functions.

• Antipatterns - An antipattern is not the same as a bug or a coding error; it's just a common approach that causes more problems than it solves.

1.2 JavaScript: Concepts

None-Objects: Primitive types - number, string, boolean, null, and undefined

1.2.1 Object- Oriented

**Activation Object** which is a global object which has attributes.

**Object**: a collection of named properties, a list of key-value pairs. Some properties could be functions.

**Objects types**

1. Native

Described in the ECMAScript standard

1. Host

Defined by the host environment (for example, the browser environment, e.g. window and all the DOM object) .

Objects can also be categorized by:

1. Build-in (e.g. Array, Date).
2. User-defined (e.g. var o ={}).

1.2.2 No Classes

There are no long parent-child inheritance chains.

There are no classes and object composition is what you do anyway.

1.2.3 Prototypes

prototype is an object (not a class or anything special) and every function has a prototype property.

1.2.4 Environment

1. Browser patterns
2. Practical applications of a pattern

1.3 ECMAScript 5

**Strict mode** - for backward compatible.

function my() {

**"use strict";**

// rest of the function...

}

This means the code in the function is executed in the strict subset of the language. For older browsers this is just a string not assigned to any variable, so it's not used, and yet it's not an error.

In this sense ES5 is a transitional version—developers are encouraged, but not forced, to write code that works in strict mode.

**Principle on writing code under strict mode**

• Ensuring the offered code samples will not raise errors in strict mode

• Avoiding and pointing out deprecated constructs such as arguments.callee

• Calling out ES3 patterns that have ES5 built-in equivalents such as Object.create()

1.4 [JSLint](http://jslint.com/) - A kind of tool for grammar check.

1.5 The console - [fire bug](https://getfirebug.com/)