ECMAScript 是 JAVAScript 的官方命名，JAVAScript是一种编程性语言。

1. JS的基本语法原则

注意等号的两种不同用法：一个单独的等号，用于为变量赋值。三个连续的等号，用于比较两个值。

语句和表达式：

分号：用于结束语句，而不是结束块

注释：单行注释和多行注释 // /\* \*/

1. 变量和赋值

Js的变量在声明后使用。

赋值：变量声明和赋值可以同时进行，也可以为一个已有变量进行赋值。

复合赋值运算符：

标识符与变量名：大致来讲，标识符的第一个字符可以是任意Unicode字符，美元符，或者下划线，后面的字符除此之外还可以为任意的Unicode数字。有一部分标识符为保留字，他们是语法的一部分，不能作为变量名使用。

1. 值

JS 中所有的值都有属性，每一个属性都有一个key（或是name）和一个value。通过（.）操作符可以读取属性。

原始值和对象：原始值包括布尔值、数字、字符串、null、undefined；其他的值都是对象。这两者之间最主要的区别是它们的比较方式：每一个对象都有唯一的标识且只（严格的）等于自己；相反，所有的原始值，只要编码值相同，则被认为相同。

原始值：5个

原始值具有一下特点：按值进行比较（内容比较）；不可改变（其属性不能被改变、添加或移除），其中读取未知属性的时候，总会返回undefined；

对象：所有的非原始值都是对象。

做常见的对象有：简单对象：可以通过对象字面量来创建；数组：可以通过数组字面量来创建；正则表达式：可以通过正则表达式字面量来创建。

对象具有以下特点：按引用进行比较（比较身份标识，每个值都有各自的身份标识）；默认可变（对象属性可以很自由的被改变、添加和移除）

Undefined和null：前者的意思是没有值，即未被初始化的变量，另外丢失的参数、访问不存在的属性也会是undefined；后者的意思是没有对象，在用到对象的时候他表示空值。两者都没有属性，甚至连toString（）这种标准方法都没有，都可被视为false。

使用typeof和instanceof对值分类：前者主要用于原始值，后者用于对象。

1. 布尔值

原始布尔类型包含true和false两个值。

真值和假值：以下的会被解释成false🡪undefined、null、false、0、NaN、““；

其余的都是true。

二元逻辑运算符：

等式运算符：

1. 数字

Js中所有的数字都是浮点数。

也包含一些特殊的数字，NaN：一个错误的值；

Infinity：多数情况下也是一个错误的值。它比任何一个数都要大NaN除外；

1. 运算符
2. 字符串

字符串可以直接通过字符串字面量来创建。这些字面量限定在单引号或双引号之间。反斜杠（\）用于转义字符及产生一些控制字符……

字符串运算符：字符串可以通过加号进行连接，如果其中一个运算符是字符串的话，另一个运算数将被转换为字符串……

字符串方法：

1. 语句

条件语句：

循环语句：

有两条语句适用于所有的循环方式：break可以跳离循环，continue会开始一个新的循环迭代。

1. 函数

可以通过函数声明的方式来定义函数……

函数声明的提升特性：它们的实体会被移动到所在作用域的开始处。这使得我们可以引用后面声明的函数。

注意 var声明也具有提升的特性，但通过他们执行的赋值却不具备该特性。？？

特殊的变量arguments：在js中，函数的所有参数可以被自由调用，它会通过arguments变量来使所有参数可用。Arguments看起来像个数组，但却不具备数组的方法。

参数太多或太少：……

可选参数：

强制参数长度：

将arguments转换为数组：arguments不是数组，它只是类似于数组。它有length属性，可以通过方括号去访问它的元素。不能移除它的元素，也不能对它调用数组的方法。因此，有时候会需要将它转换为数组。

1. 异常捕获：使用try语句包裹关键代码，如果try语句有异常会被抛出那么catch语句就会执行。
2. 严格模式：严格模式激活更多的警告以及使js变得更干净。要切换到严格模式，在js文件或者<script>标签第一行输入：……
3. 变量作用域和闭包：

在js中，通过在变量前使用var语句声明变量；可以使用单个var语句声明和初始化多个变量；由于前置的缘故，通常它的最佳实践是在一个函数的开始部分声明变量。

变量是函数作用域的：一个变量的作用域总是完整的函数（相对于当前块）

变量的提升特性：所有的变量声明都会被提升，声明会被移动到函数的开始处，而赋值则仍然会在原来的位置进行。

闭包：每个函数都和它周围的变量保持着联系，哪怕它离开被创建时的作用域也是如此。

IIFE模式：引入一个新的作用域。

有时，你会想引入一个新的作用域，例如，防止一个变量变成全局变量。在js中，不能通过块来做，必须使用函数。不过有一种模式可以将函数当作类似块的方式来使用。被称为IIFE。

IIFE用例：闭包造成的无意共享。例如闭包会持续地保持与外部变量的连接，而这有时候并不是你想要的，所以就可以应用IIFE。

1. 对象和构造函数

Js两种基础的面向对象机制：单一对象和构造函数

单一对象：……使用in运算符检查属性是否存在，如果读取一个不存在的属性，会得到undefined……；使用delete运算符移除属性。

任意属性名：属性的键名可以是任何字符串。迄今为止，我们见到过对象字面量中的属性名和点运算符后的属性名。然后，只有当它们是标识符的时候才可以这样使用。如果想用其它的字符串作为属性名，则必须将它们用引号引起来，再通过对象字面量和方括号来获取或设置这个属性。

提取方法：如果对方法进行提取，则会失去与对象的连接。

处理这个问题的解决方案可以使用bind（）方法，所有函数都支持。它会创建一个this总是指向给定值的新函数。

方法中的函数：所有的函数都有其特殊的this变量。如果在方法中有嵌套函数，这可能会不太方便，因为在嵌套函数内部不能访问方法中的this变量。

因此解决方案：第一种，我们可以将this保存在不同的变量中；第二种，利用forEach的第二个参数，它可以给this指定一个值。

函数表达式在js中通常被当作函数调用中的参数来使用。在这些函数表达式中引用this要特别的小心。

构造函数：对象工厂。除了真正的函数和方法，函数在js中还扮演了另外一个角色：如果用new运算符来调用的话，它们将变成构造函数即对象工厂。构造函数就是这样简单的模拟了其它语言的类。按照惯例，构造函数的名称以大写字母开头。

1. 数组

数组是一些有序的元素，可以通过证书索引从0开始被访问。

数组字面量：你可以通过整数索引来访问它们；length可表明数组有多少元素，可以使用它来添加或删除元素；in操作符可以在数组中正常使用；注意数组是对象，所以可以拥有对象属性。

数组方法：

遍历数组：其中最重要的方法是forEach以及map。

ForEach迭代数组并且将当前的元素和元素的index扔到一个函数中。Map通过应用一个函数映射到现有的数组的每个已经存在的元素创建一个新的数组。

1. 正则表达式：看不懂。

Js内置的支持正则表达式。它们使用斜线分割。

：test（）方法：拼配吗？；exec（）方法：匹配以及捕获分组；replace（）方法：搜索和替换

1. Math

Math是一个包含运算功能的对象。

1. 标准库的其它功能

Date（日期构造器）、JSON（解析和生成JSON数据的对象）、console.\*系列方法

第二部分：

第二章

1. Js可以自由使用吗？

Js可以说是最开放的编程语言。

1. Js优雅吗？
2. Js有用吗？

图形用户界面：

其他技术补充完善JS：有更多不是HTML5的技术对js进行着有效的补充，使得该语言变得更为实用。（类库、Node.js、JSON、NoSQL数据库）

1. Js有什么好用的工具吗？

Js正在具备越来越好的构建工具和测试工具。如Node.js使得我们可以通过Shell（不仅仅是浏览器）来运行这些工具。还有其他……

1. Js是否足够快？
2. Js是广泛使用的吗？
3. Js有前途吗？
4. 结论

第三章：js的性质

Js的本质可以概括为：

1. 它是动态的：
2. 它是动态类型：
3. 它是函数式和面向对象的：
4. 它静默失败：
5. 它部署的是开源代码：
6. 它是网络平台的一部分：
7. 古怪和非官方特性：如：明确的说，Js没有整数，只有浮点数。在内部，大多数引擎实现尽可能用整数；Js中的数组太灵活了……
8. 优雅部分：
9. 影响：Java是Js语法的参考模型，他同时影响Js将值分为原始类型和对象类型，以及日期构造函数；AWK给了Js函数的灵感，实际上关键字function来自AWK；由于Scheme，Js拥有第一类函数（函数被作为数值来对待并且可以像函数的参数一样被传递）和闭包；Js向Self借鉴了与众不同的面向对象风格，self支持对象之间的原型继承；Perl和Python影响力Js的字符串、数组和正则表达式的处理方式；HyperTalk启发了Js如何集成到浏览器，这使得HTML标签拥有事件处理属性，比如onclick。

第四章：Js是如何创造出来的。

第五章：标准化：ECMAScript

第六章：Js的历史里程碑