**第三章 基本（原始）数据类型进阶**

**Number进阶**

* Number基本数据类型变量有与其对应的Number包装对象，当访问Number基本数据类型属性或方法时创建临时包装对象，访问的都是对象中的属性或方法（注：访问对象属性时，首先访问自身属性，访问不到时，则会在原型链上寻找对应的属性和方法）

**Number原型方法(Number对象继承的方法）**- Number.prototype.toFixed(...)、Number.prototype.toPrecision(...)  
- Number.prototype.toString(...)、Number.prototype.toExponential(...)

**Number构造器属性（静态属性）**  
- Number.MAX\_VALUE、Number.MIN\_VALUE  
- Number.NaN、Number.NEGATIVE\_INFINITY、Number.POSITIVE\_INFINITY

**Number函数的使用**- 类型转换、实例化Number对象（原始值的包装对象）

**String进阶**

String基本数据类型变量有与其对应的String包装对象，当访问String基本数据类型属性或方法时创建临时包装对象，访问的都是对象中的属性或方法（注：访问对象属性时，首先访问自身属性，访问不到时，则会在原型链上寻找对应的属性和方法）

**字符串比较**- "B" > "A"、"B" > "a"  
- "B".localeCompare("A"); //考虑本地化的字符排序，返回0或非0

**字符串拼接**- 合并：加号运算符（+、+=）  
- 合并：拼接字符串数组（通过数组方法push、join等）

**String函数的使用**- 类型转换、实例化字符串对象

**字符串构造器方法（静态方法）**- String.fromCharCode(97,98,99);  
- String.fromCharCode.apply(null,[97,98,99]);

**字符串原型方法（String对象继承的方法）之提取字符串**- String.prototype.charAt(pos);  
- String.prototype.charCodeAt(pos);  
- String.prototype.slice(start,end?);  
- String.prototype.substr(start,length?)  
- String.prototype.substring(start,end?);  
- String.prototype.split(separator?,limit?);

**字符串原型方法（String对象继承的方法）之字符串变换**- String.prototype.trim( );  
- String.prototype.concat(str1?,str2?);  
- String.prototype.toLowerCase( ); String.prototype.toLocaleLowerCase( );  
- String.prototype.toUpperCase( ); String.prototype.toLocaleUpperCase( );

**字符串原型方法（String对象继承的方法）之检索和比较**- String.prototype.indexOf(searchingString,position?);  
- String.prototype.lastIndexOf(searchingString,position?);  
- String.prototype.localeCompare(other);

**字符串原型方法（String对象继承的方法）之可正则方法**- String.prototype.search(regexp);  
- String.prototype.match(regexp);  
- String.prototype.replace(regexp);

**Boolean进阶**

所有丢向都是真值

**Boolean函数的使用**

类型转换、实例化布尔对象（原始值的包装对象