1. paseInt和parseFloat函数

parseInt() 函数可解析一个字符串，并返回一个整数。

**语法**

parseInt(*string*, *radix*)

|  |  |
| --- | --- |
| **参数** | **描述** |
| string | 必需。要被解析的字符串。 |
| radix | 可选。表示要解析的数字的基数。该值介于 2 ~ 36 之间。  如果省略该参数或其值为 0，则数字将以 10 为基础来解析。如果它以 “0x” 或 “0X” 开头，将以 16 为基数。  如果该参数小于 2 或者大于 36，则 parseInt() 将返回 NaN。 |

**返回值**

返回解析后的数字。

**说明**

当参数 *radix* 的值为 0，或没有设置该参数时，parseInt() 会根据 *string* 来判断数字的基数。

举例，如果 *string* 以 "0x" 开头，parseInt() 会把 *string* 的其余部分解析为十六进制的整数。如果 *string* 以 0 开头，那么 ECMAScript v3 允许 parseInt() 的一个实现把其后的字符解析为八进制或十六进制的数字。如果 *string* 以 1 ~ 9 的数字开头，parseInt() 将把它解析为十进制的整数。

**提示和注释**

**注释：**只有字符串中的第一个数字会被返回。

**注释：**开头和结尾的空格是允许的。

**提示：**如果字符串的第一个字符不能被转换为数字，那么 parseFloat() 会返回 NaN。

1. 十进制和十六进制的数据的相互转换:

* 前面有“0X”或者“0x”，例如:

**var *node***= 0x000010;

* 方法一:parseInt(node)或者parseInt(node, 10);
* 方法二:其实，系统默认就已经将node转为十进制了，直接使用即可;
* 前面没有“0X”或者“0x”，例如:

**var *node***= 000010;

* parseInt(***node***, 16)
* 十进制🡪十六进制的字符串

**function** *NumToHexString*(num){  
 **var** str= **"0x"**;  
 str+= num.toString(16);  
 **return** (str.toUpperCase());  
}

1. 字符串和ASCII之间的转换:

思路:

借助charCodeAt()和fromCharCode()方法实现转换。

* charCodeAt()方法

**定义和用法**

charCodeAt() 方法可返回指定位置的字符的 Unicode 编码。这个返回值是 0 - 65535 之间的整数。

方法 charCodeAt() 与 charAt() 方法执行的操作相似，只不过前者返回的是位于指定位置的字符的编码，而后者返回的是字符子串。

**语法**

stringObject.charCodeAt(index)

|  |  |
| --- | --- |
| **参数** | **描述** |
| index | 必需。表示字符串中某个位置的数字，即字符在字符串中的下标。 |

**提示和注释**

**注释：**字符串中第一个字符的下标是 0。如果 index 是负数，或大于等于字符串的长度，则 charCodeAt() 返回 NaN。

* fromCharCode()方法

**定义和用法**

fromCharCode() 可接受一系列指定的 Unicode 值，然后返回一个字符串。

**语法**

String.fromCharCode(numX,numX,...,numX)

|  |  |
| --- | --- |
| **参数** | **描述** |
| numX | 必需。一个或多个 Unicode 值，即要创建的字符串中的字符的 Unicode 编码。 |

**提示和注释**

**注释：**该方法是 String 的静态方法，字符串中的每个字符都由单独的数字 Unicode 编码指定。

它不能作为您已创建的 String 对象的方法来使用。因此它的语法应该是 String.fromCharCode()，而不是 myStringObject.fromCharCode()。

* 实现:

*/\*  
 \* func:字符串转为ASCII数组  
 \* @str:要转换的字符串  
 \* @return:转换后的ASCII数组  
\*/***function** *StringToCharCodeArray*(str){  
 **var** arr= [];  
 **for**(**var** i=0; i< str.**length**; ++i){  
 arr.push(str.charCodeAt(i));  
 }  
 **return** arr;  
}

*/\*  
 \* func:字符串转为ASCII数组  
 \* @str:要转换的ASCII数组  
 \* @return:转换后字符串  
 \*/***function** *CharCodeArrayToString*(arr){  
 **var** str= **""**;  
 **for**(**var** i=0; i< arr.**length**; ++i){  
 str+=String.fromCharCode(arr[i]);  
 }  
 **return** str;  
}

**//调用**

**var *node***= **" shjdhsf"**;  
**var *array***= *StringToCharCodeArray*(***node***);  
**var *str***= *CharCodeArrayToString*(***array***);  
**console**.log(***array***.join(**"-"**)+ **"\r\n"**+***str***);

//打印结果



1. 替换字符串中的某个字符

//系统自定义函数

str = str.replace(“要替换的字符串”, “替换后的字符串”);

可以是短字符串，也可以是一个字符。

上述方法中，仅仅只能替换第一个，而不是全部，所以需要自己实现:

function ReplaceAll(str, sptr, sptr1){

while (str.indexOf(sptr) >= 0){

str = str.replace(sptr, sptr1);

}

return str;

}

1. **Widget类**

CocoStudio中所有UI控件的基类都是Widget。

addChild和addNode方法：

1. 这连个方法不同，addChild只能添加继承自Widget的控件，如Button等，而addNode可以添加不是继承自
2. Widget的空间，如Node，layer等等。
3. 而且这里这几到两个成员变量CCArray\* \_widgetChildren(用于保存Widget\*)和
4. CCArray\* \_nodes(用于保存Node),但是最终都会添加到Widget继承的的Node上。因为可以添加
5. 两种类型的UI控件，所以很多方法都需要分别处理。所以我们在使用的时经常需要区别对待。
6. Cocos 2D-X 3.X中的Layer和Widget的区别

这里说的是3.x版本，因为在2.x的版本中，Layer也是可以处理触摸屏事件的。  
1. Widget是ui方面的概念，和Layer层不同，Widget也就是窗体；  
2. Widget会拦截触摸屏事件。比如一个widget覆盖在其他元素上面，那么，触摸屏事件将不会向后面传递了。而Layer则不会。  
3. Layer一般都是全屏覆盖的，无法指定大小（除了ColorLayer）。而Widget是窗口，所以可以指定大小的。  
4. Widget多用于UI方面的显示，而Layer应该是多用于游戏场景的绘制。

1. UI工程中导出的密码框

有时候，CocosStudio工程导出的UI Json文件，可能会忘记勾选密码隐形效果(用\*代替输入的字符)，打开Json文件的passwordEnable配置改为true即可。

在c语言中，定义一个函数:

//二进制-->十进制

int TwoToTen(int value){

}

int \_tmain(int argc, \_TCHAR\* argv[])

{

getchar();

return 0;

}

//二进制-->十进制

int TwoToTen(int value);

int \_tmain(int argc, \_TCHAR\* argv[])

{

getchar();

return 0;

}

int TwoToTen(int value){

return 0;

}

控制台一闪而过

**console**.log(**"测试:"**+ ***name***.toString(2)+**" "**+***name***.toString(8)+**" "**+*TenToTwo*(***name***)+**" "**+ parseInt(*TenToTwo*(***name***), 10)+ **" "**+*TwoToTen*(*TenToTwo*(***name***)));

其中parseInT(超出Int类型限制时，会有失真的情况。);

* Chrome调试JS报错“Uncaught ReferenceError: xl\_chrome\_menu is not defined”的解决办法:

使用Chrome浏览器的开发人员工具的Console面板查看JS错误日志时，总是提示如下信息：“Uncaught ReferenceError: xl\_chrome\_menu is not defined”。是因为在Chrome浏览器中安装了迅雷下载支持插件的原因，卸载或停用之后就不会有这个错误提示了。

* WebStorm 10注册码

UserName：William   
===== LICENSE BEGIN =====   
45550-12042010   
00001SzFN0n1bPII7FnAxnt0DDOPJA   
INauvJkeVJBuE5bqLEznccE4tet6tr   
RiyoMxDK8oDY93tx!ipPyGmqYYeWxS   
===== LICENSE END =====

* 下载SourceTree

注册账户，不过该账户的注册需要跨域