中科院计算所 ICTCLAS 5.0 接口文档

http://www.ictclas.org

2010-12

Online testing can be available on http://www.ictclas.org/

目录

ICTCLAS 介绍:

中国科学院计算技术研究所在多年研究工作积累的基础上，研制出了汉语词法分析系统 ICTCLAS(Institute of Computing Technology, Chinese Lexical Analysis System)，主要功能包括中文分词；词性标注；命名实体识别；新词识别；同时支持用户词典。

ICTCLAS 的技术优势:

1.综合性能最优

分词系统能否达到实用性要求主要取决于两个因素：分词精度与分析速度，这两者相互制约，难以平衡。大多数系统往往陷入“快而不准，准而不快”的窘境。我们研制出了完美 PDAT 大规模知识库管理技术（200510130690.3），在高速度与高精度之间取得了重大突破，该技术可以管理百万级别的词典知识库，单机每秒可以查询 100 万词条，而内存消耗不到知识库大小的 1.5 倍。基于该技术，ICTCLAS 分词速度单机 996KB/s，分词精度 98.45%，API 不超过 200KB，是当前世界上最好的汉语词法分析器。

2.统一的语言计算理论框架

汉语分词牵涉到汉语分词、未定义词识别、词性标注以及语言特例等多个因素，大多数系统缺乏统一的处理方法，往往采用松散耦合的模块组合方式，最终模型并不能准确有效地表达千差万别的语言现象，而 ICTCLAS 采用了层叠隐马尔可夫模型（Hierarchical Hidden Markov Model），将汉语词法分析的所有环节都统一到了一个完整的理论框架中，获得最好的总体效果，相关理论研究发表在顶级国际会议和杂志上，从理论上和实践上都证实了该模型的先进性。

3.全方位支持各种环境下的应用开发

ICTCLAS 全部采用 C/C++编写，支持 Linux、FreeBSD 及 Windows 系列操作系统，支持 C/C++/C#/Delphi/Java 等主流的开发语言；

支持 GBK 编码分词，同时支持 UTF-8 编码和 Big5 编码分词；

支持繁体中文分词;

支持多线程分词。

4.应需而变，量身定做

所有功能模块均可拆卸组装,ICTCLAS 支持 GBK、UTF-8 等编码，同时支持 BIG5，可处理简繁体中文；支持当前广泛承认的分词和词类标准，包括计算所词类标注集 ICTPOS 3.0，北大标准、滨州大学标准、国家语委标准、台湾"中研院"、香港"城市大学"；用户可以直接自定义输出的词类标准，定义输出格式；用户可以根据自己的需求，进行量身自助式定做适合自己的分词系统。

5.国内和国际权威的公开评测、五万客户的认可

有些公司为了商业目的，关门自测，自称准确度 99.50%，没有介绍测试环境和测试方法，封闭测试或者小规模的开放测试准确度 100%都不足为奇的，ICTCLAS1.0 在国内 973 专家组组织的评测中活动获得了第一名，ICTCLAS2.0 在第一届国际中文处理研究机构 SigHan 组织的评测中都获得了多项第一名，具体的参见系统评测部分。这些都是权威机构进行大规模现场开放测试的结果，真实可信。

目前，ICTCLAS 已经向国内外的企业和学术机构颁发 50,000 多份授权，其中包括腾讯、NEC、中华商务网、硅谷动力、云南日报等企业，北京大学、清华大学、华南理工、麻省大学；同时，ICTCLAS 广泛地被《科学时报》《人民日报》海外版、《科技日报》等多家媒体报道。您可以访问 Google 进一步了解 ICTCLAS 的应用情况。

ICTCLAS 5.0 新增特性

支持多种字符编码

ICTCLAS 支持常见字符编码，包括 GB2312、GBK、GB18030、UTF-8、BIG5。用户可指定

编码类型，也可让系统自动识别。

繁体中文分词

系统支持 BIG5 编码，即支持繁体中文分词。

支持多线程调用

系统内核全新升级。支持多线程调用。

1．关于字符编码参数说明

各个接口中，编码参数的需按照如下设置，接口参数中不再详细说明。若编码不确定，系统将会自动识别编码。

1.1 c/c++中为枚举值，如下：

```
enum eCodeType {
    CODE_TYPE_UNKNOWN,      // type unknown，system will automatically detect
    CODE_TYPE_ASCII,        // ASCII
    CODE_TYPE_GB,           // GB2312,GBK, gb18030
    CODE_TYPE_UTF8,         // UTF-8
    CODE_TYPE_BIG5          // BIG5
};
```

1.2 Jni中定义为int型，分别对应如下：

（0：编码未知，系统将会自动识别）

（1：ASCII）

（2：gb2312、GBK、gb18030）

（3：UTF-8）

（4：BIG5）

2．C++接口

2.1 ICTCLAS_Init

依据指定的路径初始化系统词典和配置信息。

声明

```
bool ICTCLAS_Init(const char* pszInitDir=NULL);
```

Routine	Required Header
ICTCLAS_Init	<ICTCLAS50.h>

返回值

如果初始化成功返回 true，否则返回 false. 如初始化不成功,请查看 ictclas.log 文件了解详细错误原因.

参数

pszInitDir：初始化路径，应包含配置文件（Configure.xml）和词典目录(Data 目录)以及授权文件(user.lic). 如果这些文件及目录在系统运行当前目录下，此参数可以为 null。

说明

ICTCLAS_Init 必须在调用系统其它任何操作前进行调用,以初始化系统的基本配置信息及加载词典文件。当停止使用 ICTCLAS 后，应调用 ICTCLAS_Exit 方法清理内存缓冲区。

ICTCLAS_Init 加载失败有可能是以下原因造成:

1） 缺少系统加载所需的词典文件
系统的词典都保存在 Data 目录下，请确保初始化路径下有 Data 目录，且词典文件完整。
2） 缺少配置文件
系统配置文件名为 Configure.xml，用于保存系统所需的配置信息，请确保初始化路径下包含正确的 Configure.xml 文件。
3） 缺少授权文件 user.lic
试用授权文件名为 user.lic,缺少合法的授权文件将无法正常加载。
4） 其他未知的加载错误，请查看 ICTCLAS.log 日志文件。

示例 1

```
#include "ICTCLAS50.h"
#include <stdio.h>
#include <string.h>

int main(int argc, char* argv[])
{
    char* sResult;
    if(!ICTCLAS_Init())
    {
        printf("Init fails\n");
        return -1;
    }
    else
    {
        printf("ok\n");
    }
    const char *sParagraph ="今天参加比赛的选手有六百名";
    int nPaLen=strlen(sParagraph);
```



```
    char* sRst=0;
    int nRstLen=0;
    sRst=(char *)malloc(nPaLen*6);
    nRstLen=ICTCLAS_ParagraphProcess(sParagraph,nPaLen,sRst ,CODE_TYPE_GB,1);
    printf("%s\n",sRst);
    free(sRst);
    ICTCLAS_Exit();
    system("pause");
    return 0;
}
```

2.2 ICTCLAS_Exit

退出 ICTCLAS 分词系统，释放词典占用的系统内存空间。

声明

```
bool ICTCLAS_Exit( );
```

Routine	Required Header
ICTCLAS_Exit	<ICTCLAS50.h>

返回值

成功返回 true；否则返回 false。

说明

ICTLCAS_Exit 此方法应该在您的应用程序退出前调用，以便释放词典所占用的内存空间。

此方法会清空 ICTCLAS 占用的词典内存空间，清除临时缓冲区及其他系统资源。

如果您需要再次调用 ICTCLAS 进行分词，请重新调用 ICTLCAS_Init()加载基本信息。

示例

参见示例 1。

2.3 ICTCLAS_ImportUserDictFile

Import user-defined dictionary from a text file.

```
ICTCLAS_API unsigned int ICTCLAS_ImportUserDictFile(const char*
pszFileName, eCodeType codeType=CODE_TYPE_UNKNOWN );
```

Routine	Required

	Header
ICTCLAS_ImportUserDictFile	<ICTCLAS50.h>

Return Value

The number of lexical entry imported successfully

Parameters

pszFileName: Text filename for user dictionary

codeType: Character encoding type

Remarks

The **ICTCLAS_ImportUserDict** function works properly only if **ICTCLAS_Init** succeeds.

The text dictionary file foramt see User-defined Lexicon.

You only need to invoke the function while you want to make some change in your customized lexicon or first use the lexicon. After you import once and make no change again, ICTCLAS will load the lexicon automatically if you set UserDict "on" in the configure file. While you turn UserDict "off", user-defined lexicon would not be applied.

Example 2

```
#include <string.h>
#include <stdlib.h>
#include <string.h>
#include "ICTCLAS50.h"

int main(int argc, char* argv[])
{
  if(!ICTCLAS_Init())
    {//初始化失败，退出。
        printf("ICTCLAS INIT FAILED!\n");
        system("pause");
        return -1;
    }
```



```
  char* sSentence="1989年春夏之交的政治风波年政治风波小时降雪量小时降雨量计划
  ABC防护训练APEC会议BB机BP机C2系统C3I系统C3系统C4ISR系统C4I系统CCITT建
  议";
    int nPaLen=strlen(sSentence);
    char* sRst=0;//用户自行分配空间，用于保存结果;
    sRst=(char *)malloc(nPaLen*6);//建议长度为字符串长度的6倍。
    int nRstLen=0;
    nRstLen=ICTCLAS_ParagraphProcess(sSentence,nPaLen,sRst,CODE_TYPE_GB,1);
    printf("Before Adding User-defined lexicon, the result is:\n%s\n",sRst);

    unsigned int
nItems=ICTCLAS_ImportUserDict("userdict.txt",CODE_TYPE_GB);
    ICTCLAS_SaveTheUsrDic();//保存用户词典
    printf("%d user-defined lexical entries added!\n",nItems);
    nRstLen=ICTCLAS_ParagraphProcess(sSentence,nPaLen,sRst,CODE_TYPE_GB,1);
    printf("After Adding User-defined lexicon, the result is:\n%s\n",sRst);
    free(sRst);

    //退出.
    //释放相关资源
    ICTCLAS_Exit();
    return ;
}
```

```
Before Adding User-defined lexicon, the result is:
1989年/t 春/tg 夏/tg 之/uzhi 交/ng 的/udel 政治/n 风波/n 1989年/t 政治/n 风
波/n
24/m 小时/n 降雪/vn 量/n 24/m 小时/q 降雨量/n 863/m 计划ABC防护训练APEC会议BB
机B
P机C2系统C3I系统C3系统C4ISR系统C4I/nt 系统/n CCITT/x 建议/n
14321 user-defined lexical entries added!
After Adding User-defined lexicon, the result is:
1989年春夏之交的政治风波/n 1989年政治风波/n
24小时降雪量/n 24小时降雨量/n 863计划/n ABC防护训练/vn APE
C会议/nz BB机/n BP机/n C2系统/n C3I系统/n C3系统/n C4ISR系统/n C4I系统/n
CCITT建议/t
```

2.4 ICTCLAS_ImportUserDict

Import user-defined dictionary from a text file.

```
ICTCLAS_API unsigned int ICTCLAS_ImportUserDict(
                                    const char* pszDictBuffer,
                                    const int nLength,
                                    eCodeType codeType
```


);

Routine	Required Header
ICTCLAS_ImportUserDictFile	<ICTCLAS50.h>

Return Value

The number of lexical entry imported successfully

Parameters

pszDictBuffer: 用户词汇字符串

1.词语与词性用‘@@’间隔;

2.词与词之间用 半角‘;’间隔;

3.词性可省略

例如:“中科院@@nr;分词 v;系统@@adj;……;”,

或者:“中科院;分词;系统;……;”

nLength: 字符串长度

codeType: 字符串编码

Remarks

The **ICTCLAS_ImportUserDict** function works properly only if **ICTCLAS_Init** succeeds.

The text dictionary file foramt see User-defined Lexicon.

You only need to invoke the function while you want to make some change in your customized lexicon or first use the lexicon. After you import once and make no change again, ICTCLAS will load the lexicon automatically if you set UserDict "on" in the configure file. While you turn UserDict "off", user-defined lexicon would not be applied.

Example 3


```
void main()

{


    //初始化分词组件。

    //必须调用此接口后，才能调用其它接口！

    if(!ICTCLAS_Init())

    {//初始化失败，退出。

        printf("ICTCLAS INIT FAILED!\n");

        system("pause");

        return ;

    }

    //char* sSentence="随后温总理就离开了舟曲县城，预计温总理今天下午就回到北京。以上就是今天上午的最新动态。";

    char* sSentence="连接至 G-BOOK 中心后１０号高度控制管卡夹门控灯开关机油泵进油管";

        //char* sSentence="连接至 G-BOOK 中心后１０号高度控制管卡夹门控灯开关机油泵进油管";

    int nPaLen=strlen(sSentence);

    int nRstLen=0;

    //char* sRst=0;//用户自行分配空间，用于保存结果;

    //sRst=(char *)malloc(nPaLen*6);//建议长度为字符串长度的６倍。

    //nRstLen=ICTCLAS_ParagraphProcess(sSentence,nPaLen,sRst,CODE_TYPE_UNKNOWN,true); //未导入用户字典前的处理

    //printf("Before Adding User-defined lexicon, the result is:\n%s\n",sRst);

    //free(sRst);
```



```
    //"1989年春夏之交 n##政治风波年 a##ABC防护训练 n##test ss";

    //char * pUserWords="门控灯开关@@MT;1 0号高度控制管卡夹@@EPC;门控灯开关@@MT;机油泵进油管@@MT;";

    char * pUserWords="灯开@@sss";

    int nLen=strlen(pUserWords);

    unsigned int nItems=0;

    nItems=ICTCLAS_ImportUserDict(pUserWords,nLen,CODE_TYPE_UNKNOWN);//导入用户字典

    printf("%d user-defined lexical entries added!\n",nItems);

    char* sRstl=0;

    sRstl=(char *)malloc(nPaLen*6);

    nRstLen=ICTCLAS_ParagraphProcess(sSentence,nPaLen,sRstl,CODE_TYPE_UNKNOWN,true); //未导入用户字典前的处理

    printf("After Adding User-defined lexicon, the result is:\n%s\n",sRstl);

    free(sRstl);

    //释放资源退出

    ICTCLAS_Exit();

    return ;

}
```

2.5 ICTCLAS_SaveTheUsrDic

Save the user dictionary to disk.

int ICTCLAS_SaveTheUsrDic();

Routine	Required Header
ICTCLAS_SaveTheUsrDic	<ICTCLAS50.h>

Return Value

Return 1 if save succeed. Otherwise return 0.

Parameters

Remarks

The ICTCLAS_SaveTheUsrDic function works properly only if ICTCLAS_Init succeeds.

Example

参见示例2。

Output

2.6 ICTCLAS_ParagraphProcess

Process a paragraph, and return the result buffer pointer
int ICTCLAS_ParagraphProcess(const char *sParagraph,int nPaLen, char* sResult ,eCodeType eCt,int bPOStagged);

Routine	Required Header
ICTCLAS_ParagraphProcess	<ICTCLAS50.h>

Return Value

Return the pointer of result buffer and the length of the result.

Parameters

sParagraph: The source paragraph
nPaLen: The length of the paragraph
eCodeType: The character coding type of the string
bPOStagged: Judge whether need POS tagging, 0 for no tag; 1 for tagging; default:1.
sResult: The processing results

Remarks

The ICTCLAS_ParagraphProcess function works properly only if ICTCLAS_Init succeeds.

Example 4

```
#include <string.h>
#include <stdlib.h>
```



```
#include <string.h>
#include "ICTCLAS50.h"

int main(int argc, char* argv[])
{
if(!ICTCLAS_Init())
    {
        printf("Init fails\n");
        return ;
    }
    char* sSentence="当西班牙人捧起大力神杯时，西班牙首相萨帕特罗也激动不已";
    int nPaLen=strlen(sSentence);
    char* sRst=0;//用户自行分配空间，用于保存结果;
    sRst=(char *)malloc(nPaLen*6);//建议长度为字符串长度的6倍。
    int nRstLen=0;
    nRstLen=ICTCLAS_ParagraphProcess(sSentence,nPaLen, sRst,CODE_TYPE_GB,1);
    printf("The result is:\n%s\n",sRst);
    ICTCLAS_Exit();
    return;
 }
```

Output

The result is:

当/p/西班牙/nsf/人/n/捧/v/起/vf/大力神/nz/杯/ng/时/ng/，/n/西班牙/nsf/首相/n/萨帕特罗/nrf/也/d/激动/a/不已/vi/

2.7 ICTCLAS_ParagraphProcessA

```
ICTCLAS_API LPICTCLAS_RESULT  ICTCLAS_ParagraphProcessA(
    const char*  pszText,
    int            iLength,
    int            &nResultCount, //[out]
    eCodeType  codeType=CODE_TYPE_UNKNOWN,
    bool               bEnablePOS=false
    );
```

Routine	Required Header
ICTCLAS_ParagraphProcessA	<ICTCLAS50.h>

Return Value

the pointer of result vector, it is managed by system, user cannot alloc and free it.

```
struct tagICTCLAS_Result{
  int iStartPos; //开始位置
  int iLength; //长度
  char szPOS[POS_SIZE];//词性
  int   iPOS; //词性ID
  int iWordID; //词ID
  int iWordType; //词语类型, 用户词汇? (0-否,1-是)
  int lfWeight;// 词语权重
 };
```

Parameters

pszText: The source paragraph

iLength: The length of the paragraph

codeType: The character coding type of the string

bEnablePOS: Judge whether need POS tagging, 0 for no tag; 1 for tagging; default:1.

nResultCount: The length of the processing result

Remarks

The **ICTCLAS_ParagraphProcessA** function works properly only if **ICTCLAS_Init** succeeds.

Example 5

```
#include <string.h>
#include <stdlib.h>
#include <string.h>
#include "ICTCLAS50.h"

int main(int argc, char* argv[])
{
if(!ICTCLAS_Init())
    {
        printf("Init fails\n");
        return ;
    }
    char* sSentence="当西班牙人捧起大力神杯时, 西班牙首相萨帕特罗也激动不已";
    int nPaLen=strlen(sSentence);
    char* sRst=0;//用户自行分配空间, 用于保存结果;
    sRst=(char *)malloc(nPaLen*6);//建议长度为字符串长度的6倍。
```



```
    int rstCount=0;

     t_pstRstVec stVec=ICTCLAS_ParagraphProcessA(sSentence,nPaLen,
rstCount ,CODE_TYPE_GB,g_bPOSTagged);

    for (int i=0;i<rstCount;i++)
    {//打印分词结果
       printf("start=%d,length=%d\r\n",rstVec[i].start,rstVec[i].length);
    }
    ICTCLAS_ResultFree(rstVec); //调用接口释放内存
    ICTCLAS_Exit();//释放资源退出
    return ;
}
```

Output

```
start=0,length=2

start=2,length=6

start=8,length=2

start=10,length=2

start=12,length=2

start=14,length=6

start=20,length=2

start=22,length=2

start=24,length=2

start=26,length=6

start=32,length=4

start=36,length=8

start=44,length=2

start=46,length=4

start=50,length=4
```


2.8 ICTCLAS_FileProcess

Process a text file

```
ICTCLAS_API bool ICTCLAS_FileProcess(
    const char* pszSrcFileName,
    const char* pszDstFileName,
    eCodeType  srcCodeType=CODE_TYPE_UNKNOWN,
    bool       bEnablePOS=false
);
```

Routine	Required Header
ICTCLAS_FileProcess	<ICTCLAS50.h>

Return Value

Return true if processing succeed. Otherwise return false.

Parameters

sSourceFilename: The source file path to be analysized;
sDsnFilename: The result file name to store the results.
eCodeType: The character code type of the source file
bPOStagged: Judge whether need POS tagging, 0 for no tag; 1 for tagging; default:1.

Remarks

The **ICTCLAS_FileProcess** function works properly only if **ICTCLAS_Init** succeeds.

The output format is customized in ICTCLAS configure.

Example 6

```
#include <string.h>
#include <stdlib.h>
#include <string.h>
#include "ICTCLAS50.h"

int main(int argc, char* argv[])
{
if(!ICTCLAS_Init())
{
printf("Init fails\n");
return -1;
```



```
}
ICTCLAS_FileProcess("Test.txt", "Test_result.txt",CODE_TYPE_GB,1);
ICTCLAS_Exit();
return 0;
}
```

Output: Please check the “Test_result.txt”

2.9 ICTCLAS_SetPOSmap

select which pos map will use.

```
int ICTCLAS_SetPOSmap(int nPOSmap);
```

Routine	Required Header
ICTCLAS_SetPOSmap	<ICTCLAS50.h>

Return Value

Return 1 if excute succeed. Otherwise return 0.

Parameters

Parameters :nPOSmap : ICT_POS_MAP_FIRST 计算所一级标注集
ICT_POS_MAP_SECOND 计算所二级标注集
PKU_POS_MAP_SECOND 北大二级标注集
PKU_POS_MAP_FIRST 北大一级标注集

Remarks

The **ICTCLAS_SetPOSmap** function works properly only if **ICTCLAS_Init** succeeds.

Example 7

```
#include <string.h>
#include <stdlib.h>
#include <string.h>
#include "ICTCLAS50.h"
int main(int argc, char* argv[])
{
if(!ICTCLAS_Init())
    {
        printf("Init fails\n");
```



```
        return;

    }

    ICTCLAS_SetPOSmap(ICT_POS_MAP_FIRST);

    char* sSentence="当西班牙人捧起大力神杯时，西班牙首相萨帕特罗也激动不已";

    int nPaLen=strlen(sSentence);

    char* sRst=0;//用户自行分配空间，用于保存结果;

    sRst=(char *)malloc(nPaLen*6);//建议长度为字符串长度的6倍。

    int nRstLen=0;

    nRstLen=ICTCLAS_ParagraphProcess(sSentence,nPaLen, sRst,CODE_TYPE_GB,1);

    printf("The result is:\n%s\n",sRst);

    ICTCLAS_Exit();

    return 0;
}
```

Output

```
The result is:
当/p/西班牙/n/人/n/捧/v/起/v/大力神/n/杯/n/时/n/，/n/西班牙/n/首相/n/萨帕特罗/n/
也/d/激动/a/不已/v/
```

2.10 ICTCLAS_ResultFree

bool ICTCLAS_ResultFree (t_pstRstVec pRetVec)

Routine	Required Header
ICTCLAS_ResultFree	<ICTCLAS50.h>

Return Value

Return true if excute succeed. Otherwise return false.

Parameters

t_pstRstVec: the pointer of processing result vector

Remarks

The **ICTCLAS_ResultFree**function works properly only if **ICTCLAS_Init** succeeds.The output format is customized in ICTCLAS configure.

Example

参见 Example 6

3．JNI接口

3.1 ICTCLAS_Init

Init the analyzer and prepare necessary data for ICTCLAS according the configure file.

bool ICTCLAS_Init( );

Routine	Required Header
ICTCLAS_Init	ICTCLAS.I3S.AC.ICTCLAS50

Return Value

Return true if init succeed. Otherwise return false.

Parameters

sInitDirPath: Initial Directory Path, where file Configure.xml and Data directory stored.

the default value is 0, it indicates the initial directory is current working directory path

Remarks

The **ICTCLAS_Init** function must be invoked before any operation with ICTCLAS. The whole system need call the function only once before starting ICTCLAS. When stopping the system and make no more operation, ICTCLAS_Exit should be invoked to destroy all working buffer. Any operation will fail if init do not succeed.

ICTCLAS_Init fails mainly because of two reasons: 1) Required data is incompatible or missing 2) Configure file missing or invalid parameters. Moreover, you could learn more from the log file ictclas.log in the default directory.

Example

```
import ICTCLAS.I3S.AC.ICTCLAS50;
import java.util.*;
import java.io.*;

public class ICTCLAS50 {
public static void main(String[] args)
{
            try
            {
                ICTCLAS50 testICTCLAS50 = new ICTCLAS50();
                //分词所需库的路径
                String argu = ".";
                //初始化
            if (testICTCLAS50.ICTCLAS_Init(argu.getBytes("GB2312")) ==
false)
                {
                    System.out.println("Init Fail!");
                    return;
                }
                else { System.out.println("Init Succeed!"); }

             String sInput="点击下载超女纪敏佳深受观众喜爱。
                        禽流感爆发在非典之后。";
              byte nativeBytes[] =
                testICTCLAS50.ICTCLAS_ParagraphProcess(sInput.getBytes("GB2312"
                ), 0, 1);
                System.out.println(nativeBytes.length);
```



```
                String nativeStr = new String(nativeBytes, 0,
                nativeBytes.length, "GB2312");
                System.out.println("The result is : " + nativeStr);
                testICTCLAS50.ICTCLAS_Exit();
            }
            catch (Exception ex)
            {
            }
}
}
```

Output

Init Success!
点/n 击/vg 下载/v 超女/nz 纪敏佳/nr 深受/v 观众/n 喜爱/vn 。/wj 禽流感/n 爆发/v 在/p 非典/nz 之后/f 。/wj

3.2 ICTCLAS_Exit

Exit the program and free all resources and destroy all working buffer used in ICTCLAS.

```
boolean ICTCLAS_Exit();
```

Routine	Required Header
ICTCLAS_Exit	ICTCLAS.I3S.AC.ICTCLAS50

Return Value

Return true if succeed. Otherwise return false.

Parameters

none

Remarks

The **ICTCLAS_Exit** function must be invoked while stopping the system and make no more operation. And call ICTCLAS_Init function to restart ICTCLAS.

Example

```
import ICTCLAS.I3S.AC.ICTCLAS50;
import java.util.*;
import java.io.*;
```



```
public class ICTCLAS50 {
public static void main(String[] args)
    {
            try
            {
                ICTCLAS50 testICTCLAS50 = new ICTCLAS50();
                //分词所需库的路径
                String argu = ".";
                //初始化
            if (testICTCLAS50.ICTCLAS_Init(argu.getBytes("GB2312")) == false)
                {
                    System.out.println("Init Fail!");
                    return;
                }
                else { System.out.println("Init Succeed!"); }

        String sInput="点击下载超女纪敏佳深受观众喜爱。禽流感爆发在非典之后。";
              byte nativeBytes[] =
              testICTCLAS50.ICTCLAS_ParagraphProcess(sInput.getBytes("GB2312")
              , 0, 1);
               System.out.println(nativeBytes.length);
               String nativeStr = new String(nativeBytes, 0,
               nativeBytes.length, "GB2312");
               System.out.println("The result is : " + nativeStr);
               testICTCLAS50.ICTCLAS_Exit();
            }
            catch (Exception ex)
            {
            }
}
}
```

Output

Init Success!
点/n 击/vg 下载/v 超女/nz 纪敏佳/nr 深受/v 观众/n 喜爱/vn 。/wj 禽流感/n 爆发/v 在/p 非典/nz 之后/f 。/wj

3.3 ICTCLAS_ImportUserDict

Import user-defined dictionary from a text file.

```
int ICTCLAS_ImportUserDict(byte[] sPath);
```


Routine	Required Header
ICTCLAS_ImportUserDict	ICTCLAS.I3S.AC.ICTCLAS50

Return Value

The number of lexical entry imported successfully

Parameters

sPath: Text filename for user dictionary

Remarks

The **ICTCLAS_ImportUserDict** function works properly only if **ICTCLAS_Init** succeeds.

The text dictionary file foramt see **User-defined Lexicon**.

You only need to invoke the function while you want to make some change in your customized lexicon or first use the lexicon. After you import once and make no change again, ICTCLAS will load the lexicon automatically if you set UserDict "on" in the configure file. While you turn UserDict "off", user-defined lexicon would not be applied.

Example

```
import ICTCLAS.I3S.AC.ICTCLAS50;
import java.util.*;
import java.io.*;

public class ICTCLAS50 {
    public static void main(String[] args) {
        ICTCLAS50 testICTCLAS50 = new ICTCLAS50();
        //init
        String argu="";
        if (testICTCLAS50.ICTCLAS_Init(argu.getBytes("GB2312")) == false) {
            System.out.println("Init Fail!");
            return ;
        }
        System.out.println("Init Success!");

           //导入用户词典前分词
```



```
            byte nativeBytes[] =
testICTCLAS50.ICTCLAS_ParagraphProcess(sInput.getBytes("GB2312"), 0, 1);
            System.out.println(nativeBytes.length);
            String nativeStr = new String(nativeBytes, 0, nativeBytes.length,
"GB2312");
            System.out.println("未导入用户词典: " + nativeStr);

            //导入用户字典
            int nCount = 0;
            String usrdir = "usrdir.txt"; //用户字典路径
            byte[] usrdirb = usrdir.getBytes();
            //第一个参数为用户字典路径, 第二个参数为用户字典的编码类型(0:type
            unknown;1:ASCII 码;2:GB2312,GBK,GB10380;3:UTF-8;4:BIG5)
            nCount = testICTCLAS50.ICTCLAS_ImportUserDict(usrdirb, 3);
            System.out.println("导入用户词个数"+ nCount);
            nCount = 0;


            //导入用户字典后再分词
            byte nativeBytes1[] =
testICTCLAS50.ICTCLAS_ParagraphProcess(sInput.getBytes("GB2312"), 0, 1);
            System.out.println(nativeBytes1.length);
            String nativeStr1 = new String(nativeBytes1, 0,
nativeBytes1.length, "GB2312");
            System.out.println("导入用户词典: " + nativeStr);


            //释放分词组件资源
             testICTCLAS50.ICTCLAS_Exit();
    }
}
```

Output

Init Success!
Before Import User Dictionary: 点/n 击/vg 下载/v 超女/nz 纪敏佳/nr 深受/v 观众/n 喜爱/vn 。/wj 禽流感/n 爆发/v 在/p 非典/nz 之后/f 。/wj
Import User Dictionary entries: 3
After Import User Dictionary: 点击/v 下载/v 超女/nz 纪敏佳/nr 深受/v 观众/n 喜爱/vn 。/wj 禽流感/n 爆发/v 在/p 非典/nz 之后/f 。/wj

3.4 ICTCLAS_ParagraphProcess

Process a paragraph, and return the result buffer pointer
public native byte[] ICTCLAS_ParagraphProcess(byte[] sSrc, int eCodeType, int bPOSTagged);

Routine	Required Header
ICTCLAS_ParagraphProcess	ICTCLAS.I3S.AC.ICTCLAS50

Return Value

The result of the processing.

Parameters

sSrc: The source paragraph
eCodeType: The character coding type of the string
bPOStagged: Judge whether need POS tagging, 0 for no tag; 1 for tagging; default:1.

Remarks

The **ICTCLAS_ParagraphProcess** function works properly only if **ICTCLAS_Init** succeeds.

Example

```
import ICTCLAS.I3S.AC.ICTCLAS50;
import java.util.*;
import java.io.*;

public class ICTCLAS50 {
public static void main(String[] args)
{
    try
        {
            ICTCLAS50 testICTCLAS50 = new ICTCLAS50();
            //分词所需库的路径
            String argu = ".";
            //初始化
            if (testICTCLAS50.ICTCLAS_Init(argu.getBytes("GB2312")) == false)
            {
                System.out.println("Init Fail!");
```



```
                return;
            }
            //导入用户词典前分词
                byte nativeBytes[] =
         testICTCLAS50.ICTCLAS_ParagraphProcess(sInput.getBytes("GB2312"), 0,
         1);
            System.out.println(nativeBytes.length);
            String nativeStr = new String(nativeBytes, 0, nativeBytes.length,
         "GB2312");
            //释放分词组件资源
            testICTCLAS50.ICTCLAS_Exit();
        }
        catch (Exception ex)
        {
        }
}
}
```

Output

Init Success!

点/n 击/vg 下载/v 超女/nz 纪敏佳/nr 深受/v 观众/n 喜爱/vn 。/wj 禽流感/n 爆发/v 在/p 非典/nz 之后/f 。/wj

3.5 ICTCLAS_FileProcess

Process a text file

bool ICTCLAS_FileProcess(const char *sSrcFilename,eCodeType eCt,const char *sDsnFilename,int bPOStagged);

Routine	Required Header
ICTCLAS_FileProcess	ICTCLAS.I3S.AC.ICTCLAS50

Return Value

Return true if processing succeed. Otherwise return false.

Parameters

sSourceFilename: The source file path to be analysized;
eCodeType: The character code type of the source file
sDsnFilename: The result file name to store the results.
bPOStagged: Judge whether need POS tagging, 0 for no tag; 1 for tagging; default:1.

Remarks

The **ICTCLAS_FileProcess** function works properly only if **ICTCLAS_Init** succeeds.

Example

```
import ICTCLAS.I3S.AC.ICTCLAS50;
import java.util.*;
import java.io.*;

public static class ICTCLAS50{
public static void main(String[] args)
    {
        try
        {
            ICTCLAS50 testICTCLAS50 = new ICTCLAS50();
            //分词所需库的路径
            String argu = ".";
            //初始化
            if (testICTCLAS50.ICTCLAS_Init(argu.getBytes("GB2312")) == false)
            {
                System.out.println("Init Fail!");
                return;
            }

            //输入文件名
            String Inputfilename = "test.txt";
            byte[] Inputfilenameb = Inputfilename.getBytes();

            //分词处理后输出文件名
            String Outputfilename = "test_result.txt";
            byte[] Outputfilenameb = Outputfilename.getBytes();

            //文件分词(第一个参数为输入文件的名,第二个参数为文件编码类型,第三个
            参数为是否标记词性集 1 yes,0 no,第四个参数为输出文件名)
             testICTCLAS50.ICTCLAS_FileProcess(Inputfilenameb, 2,
             2,Outputfilenameb);
        }catch (Exception ex)
         {
         }
    }
}
```

Output

3.6 ICTCLAS_SetPOSmap

select which pos map will use.

public native int ICTCLAS_SetPOSmap(int nPOSmap);

Routine	Required Header
ICTCLAS_SetPOSmap	ICTCLAS.I3S.AC.ICTCLAS50

Return Value

Return 1 if excute succeed. Otherwise return 0.

Parameters

Parameters :nPOSmap : ICT_POS_MAP_FIRST 计算所一级标注集
ICT_POS_MAP_SECOND 计算所二级标注集
PKU_POS_MAP_SECOND 北大二级标注集
PKU_POS_MAP_FIRST 北大一级标注集

Remarks

The **ICTCLAS_SetPOSmap** function works properly only if **ICTCLAS_Init** succeeds. If the input parameter is not validate, the system will choose a default one.

Example

```
import ICTCLAS.I3S.AC.ICTCLAS50;
import java.util.*;
import java.io.*;

public class ICTCLAS50 {
    public static void main(String[] args) {
     try{
        ICTCLAS50 testICTCLAS50 = new ICTCLAS50();
        //init
        String argu="";
   if (testICTCLAS50.ICTCLAS_Init(argu.getBytes(),encoding.getBytes()) ==
       false) {
            System.out.println("Init Fail!");
            return ;
        }
        System.out.println("Init Success!");
```



```
        testICTCLAS50. ICTCLAS_SetPOSmap(ICT_POS_MAP_FIRST);

        //analysis
        str = "点击下载超女纪敏佳深受观众喜爱。禽流感爆发在非典之后。";
         nativeBytes =
         testICTCLAS50.ICTCLAS_ParagraphProcess(sInput.getBytes("GB2312"),
         0,1);
         nativeStr = new String(nativeBytes,0,nativeBytes.length,"GB2312");
        System.out.println(nativeStr);
        testICTCLAS50.ICTCLAS_Exit();
    }
    Catch(Exception ex)
    {
    }
}
}
```

Output

Init Success!

点/n 击/vg 下载/v 超女/nz 纪敏佳/nr 深受/v 观众/n 喜爱/vn 。/wj 禽流感/n 爆发/v 在/p 非典/nz 之后/f 。/wj

3.7 ICTCLAS_nativeProcAPara

将分词结果转化为 stResult 结构体输出

byte[] nativeProcAPara(byte[] sSrc, int eCodeType, int bPOStagged);

Routine	Required Header
ICTCLAS_ nativeProcAPara	ICTCLAS.I3S.AC.ICTCLAS50

Return Value

The result of the process.

Parameters

sSrc: The resource paragragh
eCodeType: The character type of the resource
bPOStagged: whether using POS tag

Remarks

The **ICTCLAS_GetWordId** function works properly only if **ICTCLAS_Init** succeeds. The result of the process would be transfer in a class type. The definition of the class is like:

```
class  stResult
{
      int start; //start position,词语在输入句子中的开始位置
      int length; //length,词语的长度
      int   iPOS; //POS,词性ID
      String  sPOS;//word type词性
      int word_ID; //word_ID,词语ID
    int word_type; //Is the word of the user's dictionary?(0-no,1-yes)查看词语是否为用户字典中词语
      int weight;// word weight,词语权重
     public void setStart(int start) {
        this.start = start;
    }
    public void setLength(int length) {
        this.length = length;
    }
    public void setiPOS(int iPOS) {
        this.iPOS = iPOS;
    }
    public void setsPOS(String sPOS) {
        this.sPOS = sPOS;
    }
    public void setWord_ID(int word_ID) {
        this.word_ID = word_ID;
    }
    public void setWord_type(int word_type) {
        this.word_type = word_type;
    }
    public void setWeight(int weight) {
        this.weight = weight;
    }

}
```

Example

```
import ICTCLAS.I3S.AC.ICTCLAS50;
import java.util.*;
import java.io.*;

Public class ICTCLAS{
public static void main()
```



```
{
try
        {
            ArrayList<stResult> a1=new ArrayList<stResult>();
            String sInput = "张晓东说的确实在理";
            ICTCLAS50 testICTCLAS50 = new ICTCLAS50();
            //初始化
            String argu = ".";
            if (testICTCLAS50.ICTCLAS_Init(argu.getBytes("GB2312")) == false)
            {
                System.out.println("Init Fail!");
                return;
            }
            //处理字符串
    byte nativeBytes[] =
testICTCLAS50.nativeProcAPara(sInput.getBytes("GB2312"), 2, 1);
            //处理结果转化
            for(int i=0;i<nativeBytes.length;i++)
            {
                //获取词语在输入句子中的开始位置
                byte a[] = Arrays.copyOfRange(nativeBytes,i, i+4);
                i+=4;
                int start = byteToInt2(a);
                start = Integer.reverseBytes(start);
                System.out.print(" "+start);
                //获取词语的长度
                byte b[] = Arrays.copyOfRange(nativeBytes,i, i+4);
                i+=4;
                int length = byteToInt2(b);
                length = Integer.reverseBytes(length);
                System.out.print(" "+length);
                //获取词性ID
                byte c[] = Arrays.copyOfRange(nativeBytes,i, i+4);
                i+=4;
                int iPOS = byteToInt2(c);
                iPOS = Integer.reverseBytes(iPOS);
                System.out.print(" "+iPOS);
                //获取词性
                byte s[] = Arrays.copyOfRange(nativeBytes,i, i+8);
                i+=8;
                String sPOS = new String(s);
                System.out.print(" "+sPOS);
                //获取词语ID
                byte j[] = Arrays.copyOfRange(nativeBytes,i, i+4);
```



```
                i+=4;
                int word_ID = byteToInt2(j);
                word_ID = Integer.reverseBytes(word_ID);
                System.out.print(" "+word_ID);
                //获取词语类型，查看是否是用户字典
                byte k[] = Arrays.copyOfRange(nativeBytes,i, i+4);
                i+=4;
                int word_type = byteToInt2(k);
                word_type = Integer.reverseBytes(word_type);
                System.out.print(" "+word_type);
                //获取词语权重
                byte w[] = Arrays.copyOfRange(nativeBytes,i, i+4);
                i+=4;
                int weight = byteToInt2(w);
                weight = Integer.reverseBytes(weight);
                System.out.print(" "+weight);
                //将处理结果赋值给结构体
                stResult stR = new stResult();
                stR.setStart(start);
                stR.setLength(length);
                stR.setiPOS(iPOS);
                stR.setsPOS(sPOS);
                stR.setWord_ID(word_ID);
                stR.setWord_type(word_type);
                stR.setWeight(weight);
                al.add(stR);
            }
        }
        catch (Exception ex)
        {
            ex.printStackTrace();
        }
    }
}
```

output

4. C#接口

4.1 ICTCLAS _ ParagraphProcessAW

Process a paragraph, API for C#


```
ICTCLAS_API int ICTCLAS_ParagraphProcessAW(
    const char*       pszText,
    LPICTCLAS_RESULT  pResult,
    eCodeType         codeType=CODE_TYPE_UNKNOWN,
    bool              bEnablePOS=false
    );
```

Routine	Required Header
ICTCLAS_FileProcessAW	<ICTCLAS50.h>

Return Value

The size of the result vector and the length of the result.

Parameters

sParagraph: The source paragraph

eCodeType: Character code type

bPOSTagged: whether set POS tag, 0:no,1:yes

t_pstRstVec: the pointer of result vector

Remarks

The **ICTCLAS_ParagraphProcessAW** function works properly only if **ICTCLAS_Init** succeeds.The output format is customized in ICTCLAS configure.

Example

```
using System;
using System.IO;
using System.Runtime.InteropServices;


namespace win_csharp
{
  //字符编码类型定义
  enum eCodeType
  {
    CODE_TYPE_UNKNOWN,//type unknown
    CODE_TYPE_ASCII,//ASCII
    CODE_TYPE_GB,//GB2312,GBK,GB10380
```



```
    CODE_TYPE_UTF8,//UTF-8
    CODE_TYPE_BIG5//BIG5
  }
  [StructLayout(LayoutKind.Explicit)]
  //处理结果类型定义
  public struct result_t
  {
    [FieldOffset(0)]
    public int start;
    [FieldOffset(4)]
    public int length;
    [FieldOffset(8)]
    public int sPos;
    [FieldOffset(12)]
    public int sPosLow;
    [FieldOffset(16)]
    public int POS_id;
    [FieldOffset(20)]
    public int word_ID;
    [FieldOffset(24)]
    public int word_type;
    [FieldOffset(28)]
    public int weight;

  }

  /// <summary>
  /// Class1 的摘要说明。
  /// </summary>
  class Class1
  {
    const string path = @"ICTCLAS50.dll";

    [DllImport(path, CharSet = CharSet.Ansi, EntryPoint = "ICTCLAS_Init")]
    public static extern bool ICTCLAS_Init(String sInitDirPath);

    [DllImport(path, CharSet = CharSet.Ansi, EntryPoint = "ICTCLAS_Exit")]
    public static extern bool ICTCLAS_Exit();

    [DllImport(path, CharSet = CharSet.Ansi, EntryPoint = "ICTCLAS_ParagraphProcessAW")]
        public static extern int ICTCLAS_ParagraphProcessAW(String sParagraph,[Out,
MarshalAs(UnmanagedType.LPArray)]result_t[] result, eCodeType eCT, int bPOSTagged);

    [DllImport(path, CharSet = CharSet.Ansi, EntryPoint = "ICTCLAS_ParagraphProcess")]
```



```
        public static extern int ICTCLAS_ParagraphProcess(String sParagraph, int nPaLen,String sResult, eCodeType eCt, int bPOStagged);

    [DllImport(path, CharSet = CharSet.Ansi, EntryPoint = "ICTCLAS_ImportUserDict")]
    public static extern int ICTCLAS_ImportUserDict(String sFilename, eCodeType eCT);

    [DllImport(path, CharSet = CharSet.Ansi, EntryPoint = "ICTCLAS_SaveTheUsrDic")]
    public static extern bool ICTCLAS_SaveTheUsrDic();
    //文件分词
    [DllImport(path, CharSet = CharSet.Ansi, EntryPoint = "ICTCLAS_FileProcess")]
        public static extern double ICTCLAS_FileProcess(String sSrcFilename, String sDsnFilename, eCodeType eCt, int bPOStagged);

    /// <summary>
    /// 应用程序的主入口点。
    /// </summary>
    [STAThread]
    static void Main(string[] args)
    {
      // 系统初始化
      if (!ICTCLAS_Init(null))
      {
        System.Console.WriteLine("Init ICTCLAS failed!");
        System.Console.Read();
      }
      //导入用户字典
      ICTCLAS_ImportUserDict("userdic.txt", eCodeType.CODE_TYPE_UNKNOWN);

      String s = "abc 他说的确实在理";
      int count = s.Length;//先得到结果的词数
      result_t[] result = new result_t[count];//在客户端申请资源
      int i = 0;

      //对字符串进行分词处理
      int nWrdCnt = ICTCLAS_ParagraphProcessAW(s, result,eCodeType.CODE_TYPE_GB, 1);
      result_t r;

      //取字符串真实长度
      byte[] mybyte = System.Text.Encoding.Default.GetBytes(s);
      byte[] byteWord;
        Console.WriteLine("No  start, length,POS_ID,Word_ID, UserDefine, Word,wordtype,weight\n");
      for (i = 0; i < nWrdCnt; i++)
```



```
        {
            r = result[i];
            String sWhichDic = "";

            switch (r.word_type)
            {
                case 0:
                    sWhichDic = "核心词典";
                    break;
                case 1:
                    sWhichDic = "用户词典";
                    break;
                case 2:
                    sWhichDic = "专业词典";
                    break;
                default:
                    break;
            }

            byteWord = new byte[r.length];

            //取字符串一部分
            Array.Copy(mybyte, r.start, byteWord, 0, r.length);
            string wrd = System.Text.Encoding.Default.GetString(byteWord);

                Console.WriteLine("{0},{1},{2},{3},{4},{5},{6}\n", wrd, r.start, r.length, r.POS_id,
r.word_ID, sWhichDic, r.weight);

        }

        //文本文件分词处理
        //ICTCLAS_FileProcess("test.txt", "Input_result.txt", eCodeType.CODE_TYPE_GB, 1);
        //释放资源退出
        ICTCLAS_Exit();
        System.Console.Read();
    }
  }
}
```

