阿洲的程式教學
關於Qt、OpenCV、影像處理演算法

XML檔操作(FileStorage)

透過標記式語言，電腦之間可以傳輸各種資訊，標記式語言概分成兩類，一種像HTML，使用國際通用的名詞，不能自行創建標記，另一種像XML，可自行定義各標籤名稱的標記式語言，這邊我們使用OpenCV提供的FileStorage進行XML檔的儲存和寫入。

這邊我們示範七種資料結構的讀取和寫入，分別為int、string、int array、string array、STL map、自創類別Person、OpenCV的Mat類別。

內文索引 [隱藏]

創建**FileStorage**類別

OpenCV用FileStorage類別寫入或讀取xml檔的資料，所以操作xml檔前須先創建此類別物件，並指明是讀取還是寫入，使用完畢後關閉：

```
FileStorage fs;
fs.open(filename, FileStorage::WRITE);   //寫入數據
fs.open(filename, FileStorage::READ);    //讀取數據
fs.release();
```

操作**int**、**string**

寫入資料時，先輸入一個string結構，當作此資料的key，接著輸入相對的value。讀取資料時透過先前輸入的key，讀到目標資料，以下兩個讀取方式結果相同。

```
fs << "name" << "John";
fs << "age"  << 27;
std:: string _name;
fs["name"] >> _name;          //方法一
_name = (string) fs["name"];  //方法二
int _age;
fs["age"] >> _age;
```

操作**array**

寫入array時，一開始輸入此陣列的key名稱，輸入實際資料前，需加上"[“，完成後加上"]"，我們使用FileNode和FileNodeIterator讀取資料，用FileNode找到這個array，再用FileNodeIterator尋訪array內部所有資料。

```
fs << "hobby" << "[" << "basketball" << "swimming" << "shopping" << "]";
FileNode hobbyNode = fs["hobby"];
FileNodeIterator it = hobbyNode.begin();
std::vector<std::string> _hobby;
while(it != hobbyNode.end()){
    _hobby.push_back((string)*it);
    it ++;
}
```

操作**map**

寫入map時，一開始輸入此map的key名稱，輸入實際資料前，需加上"{“，接著分別以key-value的順序，依序輸入完map所有的資料，最後加上"}"，使用FileNode找到這個map資料，再用個別的key找到目標資料。

```
fs << "salary" << "{" << "engineer" << 1000 << "cashier" << 700 << "}";
FileNode salaryNode = fs["salary"];
std::map<std::string, int> _salary;
_salary.insert(std::make_pair("engineer",(int)salaryNode["engineer"]));
_salary.insert(std::make_pair("cashier",(int)salaryNode["cashier"]));
```

操作**Mat**

OpenCV已經幫我們封裝好Mat的讀取和寫入，當作類似int的資料結構操作即可。

```
Mat R(3,3,CV_32F,1.0);
fs << "Mat" << R;
Mat T;
fs["Mat"] >> T;
```

操作自建類別

操作自建類別時，除了要在內別內定義write和read函式，需要在類別外覆寫OpenCV原本的write和read函式，才能有和Mat類似的操作方式。

```
Person John;
John.m_age = 30;
John.m_name = "John";
fs << "guest" << John;
Person _guest;
fs["guest"] >> _guest;
```

以下程式碼為將資料儲存成一個xml檔：

```
#include <opencv2/core/core.hpp>
#include <iostream>
#include <string>

using namespace cv;
using namespace std;

class Person{
public:
    int m_age;
    string m_name;

    void write(FileStorage& fs) const {
        fs << "{" << "age" << m_age << "name" << m_name << "}";
    }

    void read(const FileNode& node){
```

```
        m_age = (int)node["age"];
        m_name = (string)node["name"];
    }
};

static void write(FileStorage& fs, const std::string&, const Person& x){
    x.write(fs);
}

static void read(const FileNode& node, Person& x, const Person&
default_value = Person()){
    if(node.empty())
        x = default_value;
    else{
        x.read(node);
    }
}

int main(){
    FileStorage fs("demo.xml", FileStorage::WRITE);

    fs << "name" << "John";
    fs << "age"  << 27;
    fs << "hobby" << "[" << "basketball" << "swimming" << "shopping" <<
"]";
    fs << "salary" << "{" << "engineer" << 1000 << "cashier" << 700 <<
"}";

    Mat R(3,3,CV_32F,1.0);
    fs << "Mat" << R;

    Person John;
    John.m_age = 30;
    John.m_name = "John";
    fs << "guest" << John;

    fs.release();
    return 0;
}
```

File Edit Format View Help

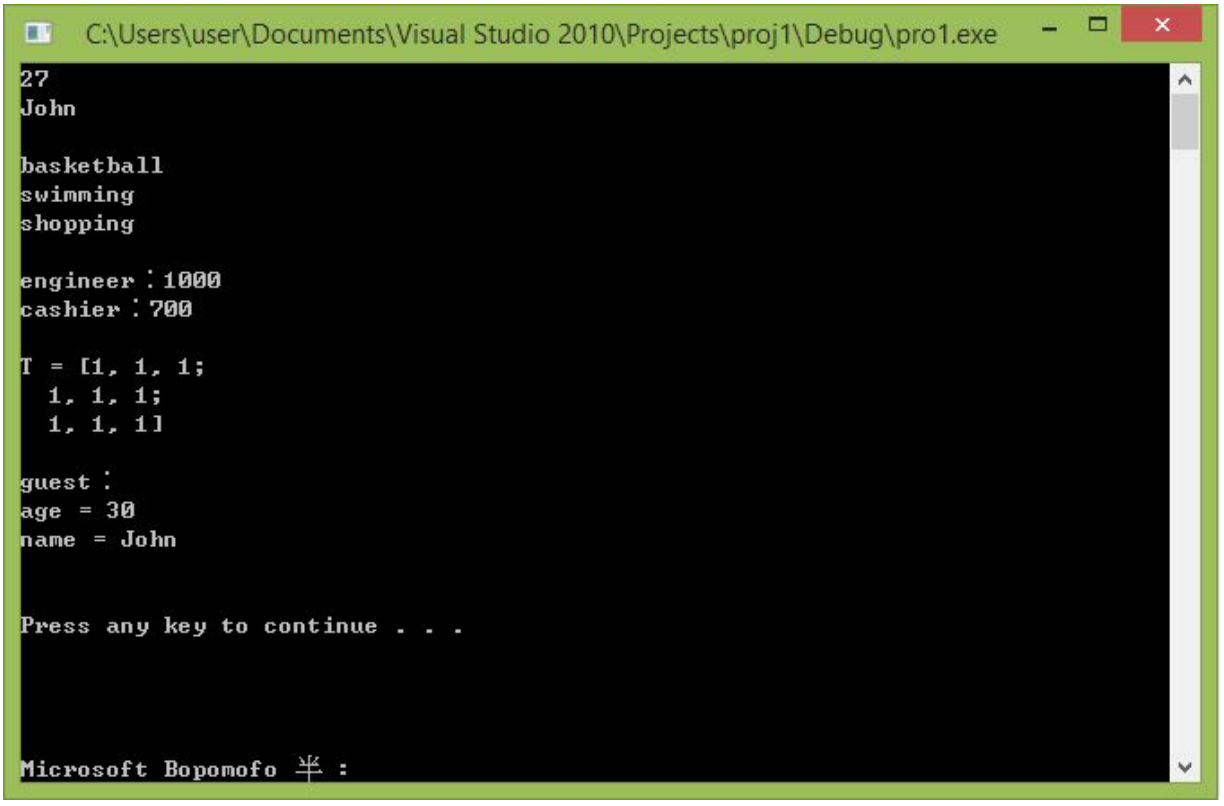
```
<?xml version="1.0"?>
<opencv_storage>
<name>John</name>
<age>27</age>
<hobby>
 basketball swimming shopping</hobby>
<salary>
 <engineer>1000</engineer>
 <cashier>700</cashier></salary>
<Mat type_id="opencv-matrix">
 <rows>3</rows>
 <cols>3</cols>
 <dt>f</dt>
 <data>
  1. 1. 1. 1. 1. 1. 1. 1. 1.</data></Mat>
<guest>
 <age>30</age>
 <name>John</name></guest>
</opencv_storage>
```

以下程式碼為讀取一個xml檔：

```
#include <opencv2/core/core.hpp>
#include <iostream>
#include <string>

using namespace cv;
using namespace std;

class Person{
public:
    int m_age;
    string m_name;

    void write(FileStorage& fs) const {
        fs << "{" << "age" << m_age << "name" << m_name << "}";
    }

    void read(const FileNode& node){
```

```
        m_age = (int)node["age"];
        m_name = (string)node["name"];
    }
};

static void write(FileStorage& fs, const std::string&, const Person& x){
    x.write(fs);
}

static void read(const FileNode& node, Person& x, const Person&
default_value = Person()){
    if(node.empty())
        x = default_value;
    else{
        x.read(node);
    }
}

static ostream& operator<<(ostream& out, const Person& m){
    out << "age = " << m.m_age << endl;
    out << "name = " << m.m_name << endl;
    return out;
}

int main(){
    FileStorage fs("demo.xml", FileStorage::READ);
    int _age;
    fs["age"] >> _age;
    std:: string _name;
    fs["name"] >> _name;
    cout << _age << endl;
    cout << _name << endl;
    cout << endl;

    FileNode hobbyNode = fs["hobby"];
    FileNodeIterator it = hobbyNode.begin();
    std::vector<std::string> _hobby;
    while(it != hobbyNode.end()){
        _hobby.push_back((string)*it);
        cout << (string)*it << endl;
        it ++;
    }
    cout << endl;

    FileNode salaryNode = fs["salary"];
    std::map<std::string, int> _salary;
    _salary.insert(std::make_pair("engineer",
(int)salaryNode["engineer"]));
    _salary.insert(std::make_pair("cashier",
(int)salaryNode["cashier"]));
    cout << "engineer：" << (int)(_salary["engineer"]) << endl;
    cout << "cashier：" << (int)(_salary["cashier"]) << endl;
    cout << endl;
```

```
    Mat T;
    fs["Mat"] >> T;
    cout << "T = " << T << endl;
    cout << endl;

    Person _guest;
    fs["guest"] >> _guest;
    cout << "guest：" << "\n" << _guest << endl;
    cout << endl;

    system("PAUSE");
    fs.release();
    return 0;
}
```



回到首頁

回到OpenCV教學

參考資料：

OpenCV 教程

2015-11-26 阿宅 OpenCV, 其他 FileStorage, OpenCV XML檔操作