C大程后端文档

> 我将把需要哪些函数，这些函数如何使用在这个文档中梳理一遍，我的注释也会简单地再说一遍

数据结构

```
typedef struct TreeNode* PtrTreeNode;

struct NodeClass{
    double lenth;
    double width;
    double dx, dy;
    int color;
};
// 跟思维导图结点有关的固有属性封装为一个结构体
struct TreeNode
{
    int NodeNumber;
    struct NodeClass NodeObject;
    char Content[50];
    PtrTreeNode FirstChild, NextSibling;
};
```

整体流程

> 前端的朋友需要自己定义一些临时变量，以方便显示和传参
>
> 我新建总是创建一个内容为空的结点，允许用户取消和编辑
>
> 另外我的逻辑是每插入一个新结点，就调用一次LevelOrderTravelsal函数(这个函数框架由我实现，但是你们需要实现一个DrawChildren的函数，我好调用画图)，实现全部从头画一遍，此时新节点内容为空，用户可以修改，你们获取用户的修改，传到一个函数里面，修改后端的存储；用户点击取消，我会在后端删除这个结点，然后你们调用LevelOrderTravelsal函数从头画一遍。
>
> 这个函数是我们绘图的核心，在里面封装了数据获取(由我实现)，图形绘制(由你们实现)

- 首先创建树(需要传入结点相关信息)

```
PtrTreeNode CreateTree(int NodeNum, struct NodeClass NodeObject, char Value[]);
```

- 判断用户鼠标是否选中我们的结点，若选中则返回指向该结点的指针，若未选中返回NULL

```
PtrTreeNode LocateNode(double x, double y, PtrTreeNode root);
```

- 编辑内容

```
void EidtContent(PtrTreeNode node, char value)
```

- 更新坐标

```
void EidtCoordinate(PtrTreeNode node, double x, double y);
```

- 删除内容(一方面是删除空结点，另一方面支持真实的级联删除)

```
void FreeNode(PtrTreeNode node);//node points to the TreeNode which is empty
void DeleteTree(PtrTreeNode subtree);//Root can't be deleted, only set content
to null
```

- 插入新结点

```
PtrTreeNode InsertTreeNode(PtrTreeNode ChosedNode, int relation, int NodeNum,
struct NodeClass NodeObject)
```

- 层序遍历(重新画一遍图，这个是我们的核心函数)

```
void LevelOrderVisit(PtrTreeNode Root);// Maybe need to combine with frontend
function.
```

- 保存为二进制文件，和读取文件

```
void Tree2File();
PtrTreeNode File2Tree();
```

所有涉及到的函数

```
// Create and initialize a tree root node or a subtree root node
PtrTreeNode CreateTree(int NodeNum, struct NodeClass NodeObject);
// To see whether the user move mouse on our node or not, (x,y) is the location
of mouse
PtrTreeNode LocateNode(double x, double y, PtrTreeNode root);
// Edit the Content
void EidtContent(PtrTreeNode node, char value);
// Edit the coordinate
void EidtCoordinate(PtrTreeNode node, double x, double y);
// Delete a node on cascade
void FreeNode(PtrTreeNode node);//node points to the TreeNode which is empty
void DeleteTree(PtrTreeNode subtree);//Root can't be deleted, only set content
to null
// Insert a Tree Node
PtrTreeNode InsertTreeNode(PtrTreeNode ChosedNode, int relation, int NodeNum,
struct NodeClass NodeObject);
void LevelOrderVisit(PtrTreeNode Root);// Maybe need to combine with frontend
function.
void Tree2File();
PtrTreeNode File2Tree();
```