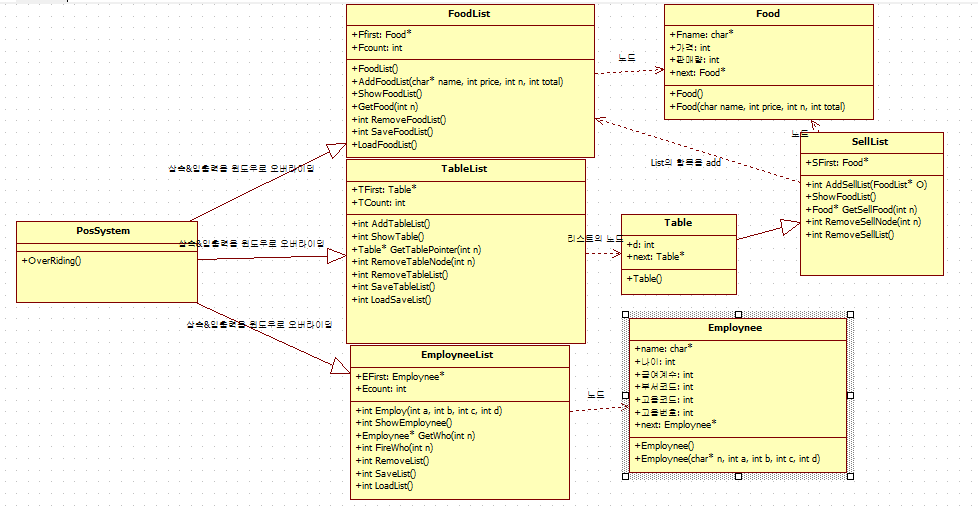
**GUI PosSystem 실체화 사양**

20120302 김우진

**시스템 구성과 class구조도**



작업환경

C++사용과 windows기반의 프로그래밍(MFC사용).

설명

Class XX – 필요 정보를 저장하며 리스트의 노드가 될 클래스

Class XXList – Class PosSytem을 이루는 한 부분. XX 클래스의 리스트를 갖게 될 클래스이며 리스트관리를 해준다. 기본적으로 출력, 입력부분이 필요한 메서드들은 CUI기반으로 미리 정의해두고 PosSystem에서 재정의한다

Class PosSystem – List 계열의 클래스들을 상속 받으며 MFC의 기본제공 객체에 상속or 멤버변수로 들어갈 예정.

**Class FoodList**

개요: Food 객체의 리스트를 보유, 관리하는 class

<설명>

Food class의 포인터에 리스트를 만들고 내부 메서드를 통해 필요한 관리를 함.

추가, 삭제, 검색, 저장, 갱신(읽어오기)를 담당하며 레스토랑의 메뉴를 담당한다.

**Class SellList**

개요: Food 객체의 리스트를 보유, 관리하는 class

설명: 테이블에서의 주문 내역을 저장하는 List. 역할은 FoodList와 동일하나 FoodList로부터 정보를 받아 메뉴를 추가하며 리스트가 삭제될 때에는 영수증이 발행된다.

**Class Table**

개요: SellList를 상속받고 테이블의 상태(사용가능, 사용 중)을 저장

<설명>

레스토랑에서 손님을 받는 단위이며 테이블의 경우에는 주문내역이 필요하여 SellList를 상속받고 사용가능한지 아닌지를 저장하는 변수를 갖는다.

**Class TableList**

개요:Table을 관리하는 class

<설명>

Table을 노드로 하는 리스트를 갖고 있으며 기본적으로 FoodList와 비슷하게 테이블을 관리한다.

**Class Employnee**

개요: 사원의 기본적인 정보를 담고 있다.

<설명>

이름과 나이를 저장하며 코드를 통하여 부서, 직급 등을 알 수 있다

**Class EmployLsit**

개요:. Employnee를 노드로 하는 리스트를 갖고 관리함

<설명>

사원을 고용,해고하며 조회한다. 이 때 코드를 통하여 여러 조회를 지원한다.

**Class PosSystem**

개요: 위의 EmployList, TableList, FoodList를 상속받아서 pos 시스템을 운영

<설명>

이 class는 MFC의 Doc클래스나 같은 MFC의 View 클래스에 상속or 내부 멤버로 들어갈 예정이며 상속을 하게 되면 오버라이딩을 통하여 입출력을 제어하며 내부 멤버로 사용된다면 일부 메서드를 삭제하고 MFC 클래스 내에서 정의하는 방식으로 사용될 것이다.

**모듈설계**

**스택 –** 기본적인 MFC 모듈 + Food, Table, Employ

**Food:** Food class에 1차적으로 종속적인 모든 클래스가 존재(FoodList,SellList)

**Table:** Table class에 종속적인 클래스가 존재(TableList)

**Employ:** Employnee class 종속적인 클래스가 존재(EmployList)

**주요 알고리즘과 자료구조**

개요: XXList 계열의 class로 이루어진 PosSystem 객체는 특별한 자료구조나 알고리즘을 요구하지 않고

XXList계열의 class는 자료구조의 단 반향 리스트를 사용하며 삽입, 삭제, 검색, 출력의 기초적인 자료구조 알고리즘을 사용한다.

**추가 알고리즘**

Int List() {

If(!this->Fisrt){ Fisrt = new List() return 0;}

List\* p= First;

While(p->next){ p=p->next}

p->next = new List();

}

**출력알고리즘**

Int showList(){

List\* p= this->First;

While(p){cout<<p;}

}

**검색 알고리즘**

Lsit\* Get(int n){

List\*p =this->Fisrt;

For(int i=n; i>0;i--) p=p->next;

Return p;

}

**삭제 알고리즘**

Int RemoveNode(int n){

P1=Get(n);

If(n=0) { this->First=p1->next; delete p1; }

Elseif(!p1->next) { P2=Get(n-1); delete p1; p2->next=null; }

Else{ P2=Get(n-1); ; p2->next=p1->next; delete p1 }

}

**전체적인 그림**

