키오스크 메뉴판 With Unity3D

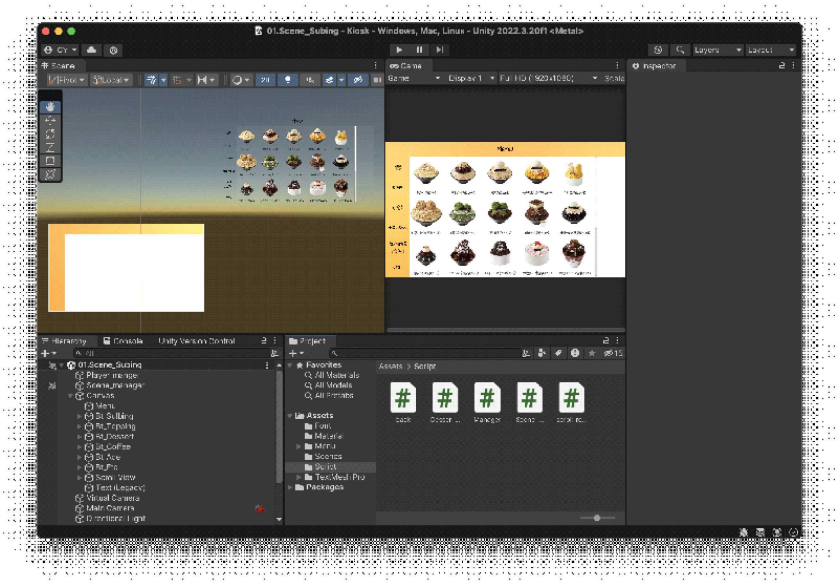
By ChoiJunYoung

현재 카페 운영 중에 메뉴판에 대한 불편함을 느낀 적이 많았다. 단종 및 신메뉴 출시, 시즌별 메뉴, 품절 등의 이유로 고객에서 정확한 메뉴 정보를 제공하기에 한계가 있었으며 현재 판매중인 업장의 메뉴 현황을 고객에게 제공하기 위해 키오스크 메뉴판을 제작하려고 한다. 함께 게임 엔진인 Unity3D를 사용해보며 엔진의 사용법과 구조를 이해하며 빌드 및 설치 단계까지 진행해 보려고 한다.

진행 과정

1) 유니티 UI(Canvas)를 통해 기능 구현(씬 전환시 에도 유지), 오브젝트를 통해 배경 지정
2) 각 메뉴 카테고리를 정해 씬 전환 설계 - Scene_Manager
3) 메뉴 클릭 시 메뉴의 정보 저장 및 출력(Name, Price, Count) - Game_Manager
4) 메뉴 삭제 시 해당 정보 삭제 및 배열의 재정렬
5) 주문할 메뉴의 정보와 최종 가격 출력

1. 기본 UI 및 오브젝트 배치(1920x1080)



Canvas안 각 씬 전환을 위한 Button과 Text파일 배치(사진과 메뉴이름=>Vertical, 메뉴 리스트(5개)=>Horizontal로 그룹화), 메뉴판 출력을 위한 Scroll1과 메뉴 장바구니 출력을 위한 Scroll2 배치

2. 기능 구현(Game_Manager)

2-1 가격과 이름 가져오기

클릭한 메뉴에 대한 괄호 안 가격과 이름을 가져오는 함수 정의

```
    public static int Get_Price()
    {
        GameObject clickObject = EventSystem.current.currentSelectedGameObject;
        GameObject Prt = clickObject.transform.parent.gameObject;
        string temp = Prt.transform.GetChild(1).GetComponent<Text>().text;
        int a = temp.IndexOf("(");
        int b = temp.IndexOf(")");
        string price = temp.Substring(a + 1, b - a - 1);
        string Price = price.Replace(".", "");
        return (int.Parse(Price)*100);
    }
    public static string Get_Name()
    {
        string ButtonName = EventSystem.current.currentSelectedGameObject.name;
        return (ButtonName);
    }
```

2-2 버튼 이벤트

고객이 선택한 메뉴를 담을 result배열 활용, 클릭 시 배열에 추가, 이미 저장된 메뉴일 경우 메뉴의 개수를 추가, 이후 메뉴의 장바구니 리스트와 결과 값을 출력

```
    public static void Button_Event()
    {
        GameObject bt_obj = GameObject.FindGameObjectWithTag("Top_list");
        string name = Get_Name();
        int price = Get_Price();
        int count=0;
        for (int i = 0; i < 30; i++)
        {
            if (result_name[i] != null) count++;
        }
        int index_arr = Array.FindIndex(result_name, element => element == name);

        if (index_arr>-1)
        {
            result_info[index_arr, 1]++;
            number = count;
        }
        else
        {
            result_name[number] = name;
            result_info[number, 0] = price;
            result_info[number, 1] = 1;
            bt_obj.transform.GetChild(number).GetChild(1).gameObject.SetActive(true);
            number++;

        }

        Show_Cal(number+1);
        Debug.Log((number) + "종류 ");
        Calculate();
        showing(number);
    }
```

2-3 삭제 이벤트

메뉴 삭제 버튼 클릭 시 위치의 배열 요소 삭제 후 출력, null오류로 인해 예외 값 설정=>배열이 아닌 리스트를 통해 재설계 필요

```
105         public static void Button_Delete()
106         {
107             GameObject bt_obj = GameObject.FindGameObjectWithTag("Top_list");
108             GameObject clickObject = EventSystem.current.currentSelectedGameObject;
109             string num_temp = clickObject.transform.GetChild(0).GetComponent<Text>().text;
110             int num = int.Parse(num_temp);//버튼 위치
111             if (result_name[num] == null) return;
112             if (num == 0)
113             {
114                 if (result_info[num, 1] > 1) result_info[num, 1]--;
115                 else
116                 {
117                     result_name[0] = null;
118                     result_info[0, 1] = 0;
119                     for (int i = 0; i < number; i++)
120                     {
121                         if (result_name[i + 1] == null)
122                         {
123                             result_name[i] = null;
124                             break;
125                         }
126                         result_name[i] = String.Copy(result_name[i + 1]);
127                         result_info[i, 0] = result_info[i + 1, 0];
128                         result_info[i,1] =result_info[i+1,1];
129                         result_info[i + 1, 1] = 0;
130                     }
131                     number--;
132                     bt_obj.transform.GetChild(number).GetChild(1).gameObject.SetActive(false);
133 
134                 }
135             }
136             else
137             {
138                 if (result_info[num, 1] > 1) result_info[num, 1]--;
139                 else
140                 {
141                     result_name[num] = null;
142                     result_info[num, 1] = 0;
143                     for (int i = num; i < number + 1; i++)
144                     {
145                         if (result_name[i + 1] == null)
146                         {
147                             result_name[i] = null;
148                             break;
149                         }
150                         result_name[i] = String.Copy(result_name[i + 1]);
151                         result_info[i, 0] = result_info[i + 1, 0];
152                         result_info[i, 1] = result_info[i + 1, 1];
153                         result_info[i + 1, 1]=0;
154                     }
155                     number--;
156                     bt_obj.transform.GetChild(number).GetChild(1).gameObject.SetActive(false);
157                 }
158             }
159             Show_Cal(number+1);
160             if (number < 0) number = 0;
161             Debug.Log(number + "종류_후");
162             Calculate();
163             showing(number);
164         }
```

2-4 기타 사용함수

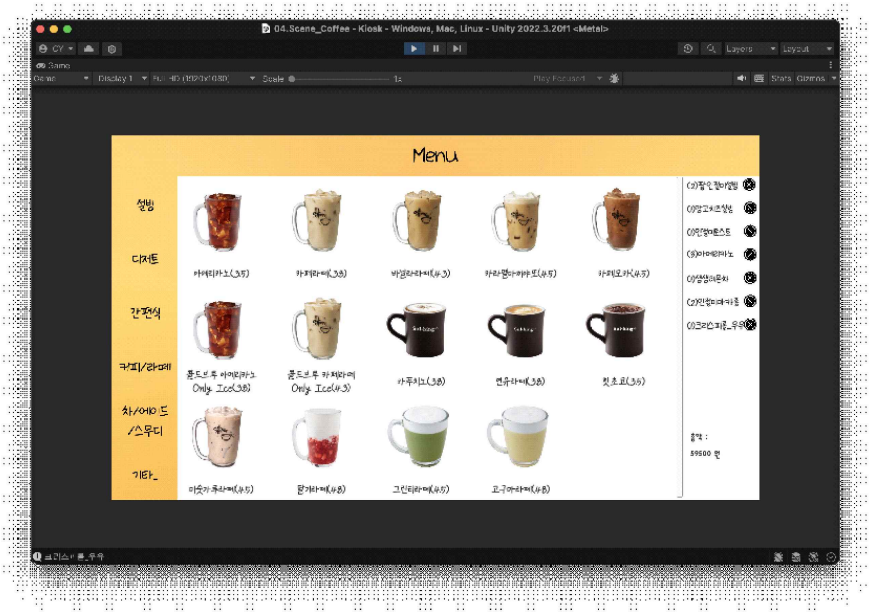
메뉴 정보 출력 및 계산 함수, 디버깅

```
166         public static void Show_Cal(int num)
167         {
168             for (int i = 0; i < num; i++)
169             {
170                 if (result_info[i, 1] < 1)
171                 {
172                     obj.transform.GetChild(i).GetChild(0).GetComponent<Text>().text = " " + result_name[i];
173                 }
174                 else
175                 {
176                     obj.transform.GetChild(i).GetChild(0).GetComponent<Text>().text = " (" +result_info[i,1]+")"+ result_name[i];
177                 }
178             }
179             Calculate();
180         }
181 
182         public static void Calculate()
183         {
184             int sum = 0;
185             for(int i = 0; i < number; i++)
186             {
187                 sum += result_info[i, 0] * result_info[i, 1];
188             }
189             GameObject sum_obj = GameObject.FindGameObjectWithTag("Sum_Result");
190             sum_obj.transform.GetComponent<Text>().text = "총액 :" + System.Environment.NewLine + sum+" 원";
191 
192         }
193 
194         public static void showing(int num)
195         {
196             for(int i = 0; i < num; i++)
197             {
198                 Debug.Log(result_name[i]);
199             }
200         }
```

3. 씬 추가 및 캔버스 구성

각 씬 마다의 카테고리 메뉴 UI 구성, Scene_Mager를 통해 각 씬 전환 설계

4. 결과 화면



보완점 및 자제평가

리소스 절약을 위한 UI구성 및 알고리즘 개선 필요, 결제를 위한 서버 구성 필요, WebGL로 빌드 시 해상도에 따른 ui 재구성 필요(Nas서버 사용함)