**돈부리 퀘스트 기술 문서**

**돈부리 퀘스트 프로젝트에 대한 간단한 설명**

**돈부리 퀘스트는 “카드게임 X 요리”를 테마로 하고 있습니다. 기존의 요리 게임들의 시스템이 실제 조리법과의 연관성이 부족하고 그 결과 요리의 테마가 다름에도 비슷한 방식의 게임 플레이를 하게 되는 부분에 초점을 맞춰 고안했습니다. 기술 문서이므로 패, 덱 등의 부분에 대한 설명은 분량 상 생략하고 있으며 이는 일반적인 카드 게임의 개념과 대체로 유사합니다. 또한 턴의 흐름, 눈으로 보이는 기본 시스템과 같이 영상에서 어느 정도 정보를 얻을 수 있는 부분 또한 생략하게 되었습니다. 알파벳이 붙어 있는 부분의 경우 해당 기능에 대한 기술적인 부분이 뒤에 기재되어 있습니다.**

**- 실제 조리법을 간소화 시킨 스테이지에서 플레이어는 요리를 완성하기 위해 “조리 기구”, “재료”, 조리 동작을 의미하는 “조리”, 그리고 요리의 특성을 나타낸 “유틸” 카드를 사용해 게임을 진행하게 됩니다. 돈부리 퀘스트의 스테이지에서는 요구하는 요리들을 주어진 턴 수 안에 완성해야 합니다. 프로토타입 영상(깃허브에 푸쉬되어 있습니다)에서는 한 종류의 요리를 하나만 만들고 있습니다. (A)**

**- 유저가 조리 기구, 재료, 그리고 일부 유틸 카드를 사용하면 4x4 필드의 네모난 칸 하나(이후 “그리드”로 칭함)에 왼쪽 위부터 차례대로 배치됩니다. 그리드에 배치된 물체들을 “오브젝트”라고 칭합니다. 그리드에 오브젝트를 배치하지 않는 카드를 “스펠” 카드라고 합니다. (B)**

**- 모든 카드는 앞면으로 사용하거나 뒷면으로 사용할 수 있으며 앞면으로 사용할 시 기존의 기능대로, 뒷면으로 사용할 시 각 카드와 연결된 “도토리 카드”로 바뀌어 패에 더해지게 됩니다. 도토리 카드는 그리드에 배치하거나 그렇지 않을 수 있습니다. 도토리 카드의 앞면은 일반적으로 특정 그리드를 메워 해당 그리드에 조리 기구와 재료가 배치되지 않기 위해 사용됩니다. 이 방식으로 그리드에 배치된 물체들을 “도토리 오브젝트”라고 칭합니다. (B)**

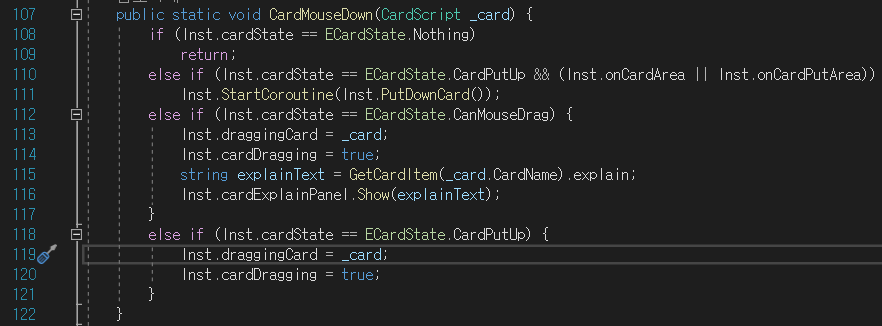
**- 도토리 카드를 뒷면으로 사용하게 되면 각 카드와 연결된 일반 카드로 바뀌어 덱에 섞여 들어간 뒤 한 장을 뽑게 됩니다. 모든 도토리 카드들의 뒷면 활용은 이 방식을 따르게 됩니다. (B)**

**- 요리 과정은 크게 “밥 짓기”와 “요리 만들기”로 나누어집니다. 밥을 짓기 위해서는 “밥솥”을 그리드에 배치한 후 “쌀”을 넣고 “불지피기”를 사용해 조리해야 합니다. 모든 요리의 조리 과정에는 “조미료”가 여러 종류 필요하며 이는 밥을 짓기 위해 필요한 카드들과 연관된 도토리 카드들인 “도토리 솥”, “도토리”, “도토리 굽기”를 사용해 조리해 얻을 수 있습니다. (C)**

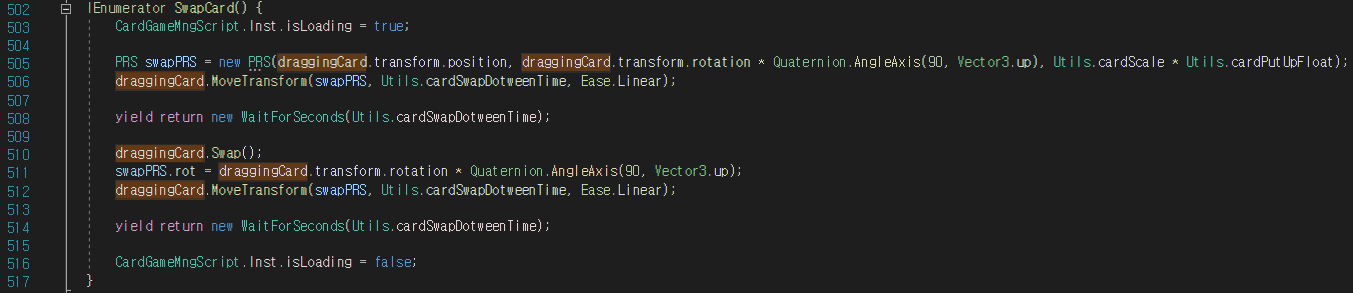
**- 특정 도토리 카드들은 특정 수의 “도토리 오브젝트”를 소모해야만 사용할 수 있습니다. 이 방식으로 도토리 카드를 사용할 시 필요 개수 만큼의 도토리 오브젝트가 사라지게 되고 비워진 그리드에 다시 오브젝트를 배치할 수 있게 됩니다. 프로토타입 영상에서는 이 기능이 아직 나타나고 있지 않습니다.**

**- 조미료 카드를 앞면으로 사용할 경우 다른 재료 카드들처럼 다음에 배치해야 할 그리드의 위치에 배치됩니다. 이 때 인접한 그리드에 조미료를 필요로 하는 오브젝트가 있으면 그 중 아직 들어있지 않은 하나를 랜덤으로 배치합니다. 없을 경우에는 “조미료” 오브젝트를 배치합니다. 조미료 오브젝트는 도토리 오브젝트처럼 기본적으로는 그리드를 메우는 역할을 하지만 인접한 그리드에 조리 도구가 있을 경우 해당 조리 도구 안으로 사라지게 되고 그리드가 비워집니다. 조미료 카드를 뒷면으로 사용하게 되면 “향기로운 도토리”가 패에 추가됩니다. (D)**

**A. 카드의 조작**

****

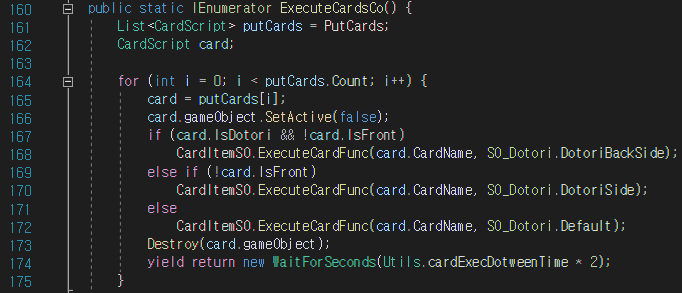
**카드의 조작은 한 장씩 이루어집니다. 카드를 드래그하게 되면 CardMngScript 내에 해당 카드의 참조가 저장되며 카드를 드래그하고 있음을 나타내는 변수인 isDragging이 true로 설정됩니다. 카드의 정보는 CardScript 클래스의 오브젝트 내에 저장되며 이는 이후에 해당 카드의 조작 시 참조됩니다.**

****

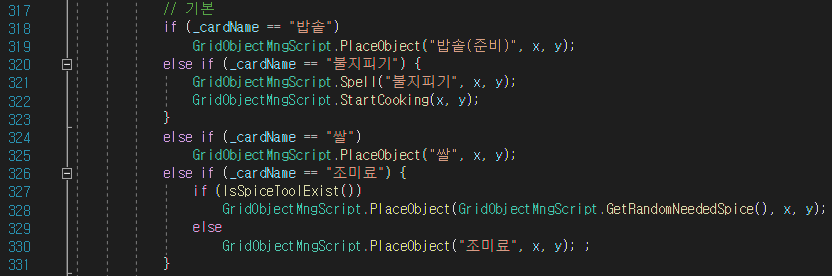
**예시로 든 Coroutine 함수는 B에서 설명하게 될 카드의 앞, 뒷면 사용을 설정하기 위해 사용됩니다. draggingCard의 transform 값을 조작하고 있는 모습입니다. SwapCard() 외에도 카드를 드래그하기 위해 사용되는 함수인 DragCard() 등에서도 draggingCard는 다수 참조되고 있습니다.**

**B. 카드의 사용과 그리드 배치**

**사용하고 싶은 카드들의 앞, 뒷면을 설정한 후에는 사용 카드 영역에 옮겨다 놓을 수 있습니다. 그 후 턴 종료를 하게 되면 사용 카드 영역에 있는 카드들이 순차적으로 사용됩니다.**

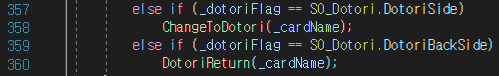
****

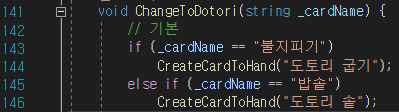
**카드들이 사용될 때 실행되는 Coroutine입니다. 각각의 카드가 Default:앞면인가, DotoriSide:뒷면인가, DotoriBackSide:도토리 카드이고 뒷면인가, 에 따라 카드의 효과 발동이 달라지게 됩니다.**

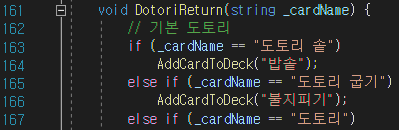
****

**Default: 일반적인 카드 사용**

**기본적으로 각각의 카드 효과에 대한 명시는 하드코딩으로 하고 있습니다. 엑셀 등의 외부 문서 프로그램을 연결해서 하는 방법도 고려하였으나 카드와 특성이 각각 가로, 세로 한 줄씩을 차지하게 되는 엑셀의 특성 상 나중에 종류가 많아지면 공백이 차지하는 부분이 많아져(모든 카드가 대부분의 특성을 가지고 있는 것이 아님) 관리가 어려울 것이라 판단하였습니다. 또한, 조건문 활용이나 코드 내 함수 호출 등의 유연한 활용이 외부 프로그램으로는 어려울 것이라 생각했습니다. 작은 함수를 많이 만들어 프로그래밍을 모르는 기획자가 작업할 때 어렵지 않도록 하는 것에 초점을 맞추었습니다.**

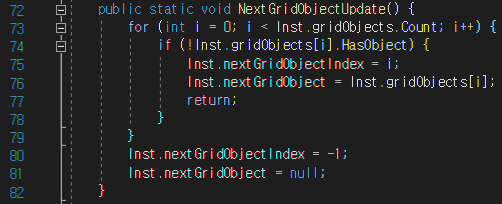
****

****

****

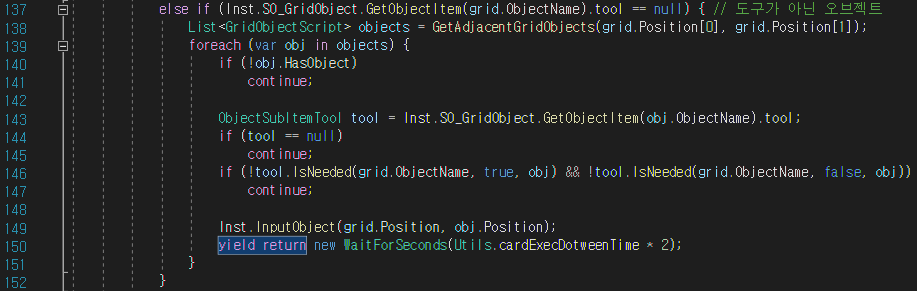
**DotoriSide, DotoriBackSide: 카드의 뒷면 사용**

**뒷면으로 사용한 카드의 효과가 발동할 때 거치는 코드입니다. 위와 마찬가지의 이유로 각각의 카드-도토리카드 의 연결을 하드코딩으로 작업하였습니다.**

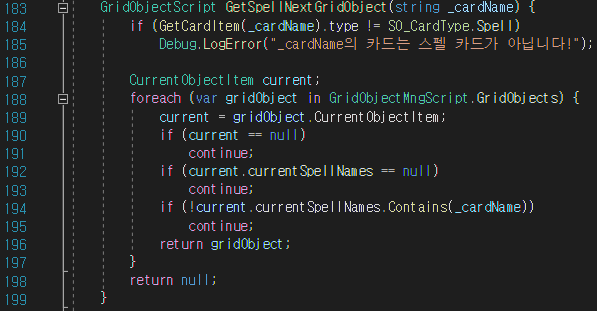
****

**그리드에 배치하는 효과를 가진 카드의 효과가 처리될 때 미리 저장해놓은 다음에 배치되어야 할 그리드의 위치 정보를 참조합니다. nextGridObject에는 그리드 자체가, nextGridObjectIndex에는 nextGridObject의 순서가 저장됩니다. nextGridObject의 정보는 그리드에 무언가가 배치되거나 사라질 때마다 갱신됩니다.**

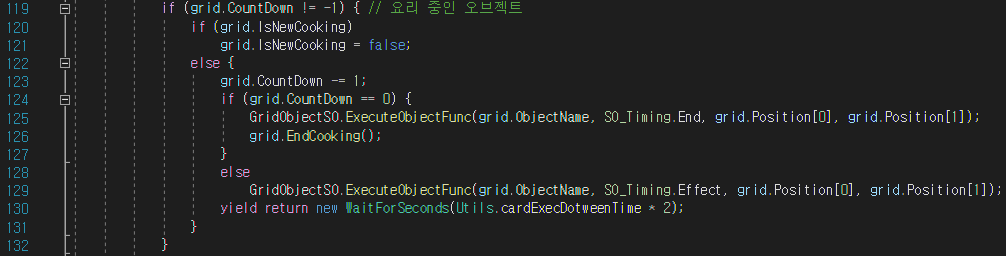
**C. 일반적인 요리 시스템과 그리드의 처리**

****

**요리를 시작하기 전에 재료를 조리 기구에 넣는 과정의 코드입니다. 그리드에 재료가 배치되어 있을 시 인접한 그리드를 탐색하여 조리 기구가 있고 해당 조리 기구가 재료를 필요로 한다면 재료를 조리 기구에 집어넣는 연출을 실행합니다. 어떤 조리 기구가 어떤 재료를 필요로 하는지에 대한 정보는 ObjectItem 클래스의 오브젝트에 저장되어 있습니다.**

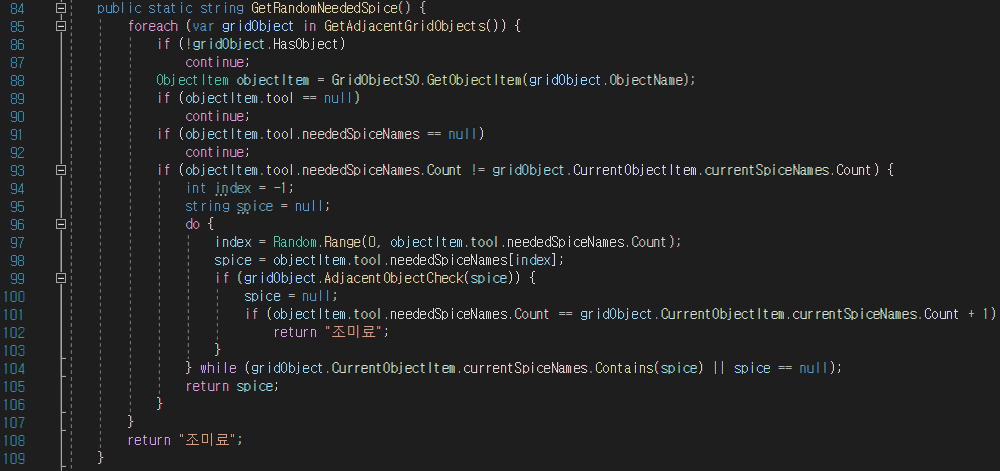
****

**스펠의 사용의 경우에는 해당 스펠이 사용될 수 있는 오브젝트가 그리드에 배치되어 있지 않으면 앞면 카드 사용이 불가하도록 원천적으로 차단합니다(드래그 후 드롭하는 것이 불가능). 카드 사용이 가능하다면, 그리드에 오브젝트를 배치하는 경우(내부에 nextGridObject를 저장)와 달리 스펠의 종류에 따라 사용 대상이 되는 오브젝트들이 다르기 때문에 처리 시점에 스펠의 대상이 될 그리드를 얻어옵니다.**

****

**조리 중일 때, 아니면 조리가 완료되었을 때 한 턴에 한번씩 호출되는 코드입니다. 조리가 완료되는 경우에는 125번 라인이 실행되는데 오브젝트가 조리 후 과정으로 넘어가는 부분은 모두 이 라인을 통해 구현됩니다. 현재 프로토타입에서는 129번 라인이 실행되지 않습니다. 그러나 추후 확장을 통해 매 턴 게임에 영향을 끼치는 오브젝트의 등장 가능성을 열어놓기 위해, 그리고 해당 기능을 추가하는 시간이 얼마 걸리지 않아 보였기 때문에 추가하게 되었습니다.**

**D. 조미료 시스템**

****

**조미료 카드가 사용될 때 실행되는 함수입니다. 함수는 결과적으로 배치될 오브젝트의 이름을 리턴합니다. 한 턴에 여러 장의 조미료 카드가 사용되고 한 개의 조리 도구의 인접한 그리드들에 두 개 이상의 조미료 카드로부터 나온 오브젝트가 배치될 예정인 경우 같은 이름의 조미료 오브젝트가 생성되지 않도록 해 주고 있습니다.**