# 〈Unity OpenAI 패키지〉

[요약]

〉 이 문서는 Unity 프로젝트에서 OpenAI(예: ChatGPT, GPT-4o/4-turbo 등)를 안전하고 재사용 가능하게 통합하는 회사용 개발 튜토리얼입니다. 목표는 빠른 PoC부터 제품화까지 이어질 수 있는 구조(UPM 패키지 사용, API 키 보호, 한글 출력, 샘플 스크립트, 배포/보안 가이드)를 제공하는 것입니다.

목차

1. 준비물 및 전제

- Unity 2020.3 LTS 이상 권장(프로젝트 정책에 맞춰 버전 조정)
- TextMeshPro 패키지(한글 렌더링용)
- OpenAI API 계정 및 Secret Key
- 회사 내부 프록시/백엔드: 키를 안전하게 보관하고 호출을 대리 수행하기 위해 권장

2. 통합 방식 선택

2.1 옵션 A — UPM 패키지(com.openai.unity)

- 장점: Unity Package Manager로 간편 설치, 예제 씬·툴 포함, OpenAI의 다양한 엔드포인트 접근성 편리
- 단점: 커뮤니티 유지 수준에 따라 업데이트 주기 다름

2.2 옵션 B — C# SDK (OpenAI-API-dotnet 등)

- 장점: 직접 제어 가능, 기존 .NET 기반 시스템과 호환성 좋음
- 단점: Unity 환경(특히 IL2CPP/NET 설정)에서 DLL 호환성 체크 필요

권장: 빠른 실험은 UPM 패키지, 서버 연동 방식이 확정된 이후 제품화 단계에서는 서버 프록시 + 안전한 SDK 조합.

## 3. OpenAI API 키 관리 (중요)

**방법 1 — 환경변수(권장 내부 테스트 시)**

- 운영체제 환경변수에 `OPENAI_API_KEY`로 저장한다.
- Unity에서 `System.Environment.GetEnvironmentVariable("OPENAI_API_KEY")`로 읽는다.

**방법 2 — 프로젝트 리소스 파일(주의)**

- 'Assets/Resources/OpenAIConfiguration' 같은 에셋에 키를 넣는 방식은 협업/버전관리 시 유출 위험이 있으므로 권장하지 않음.

**방법 3 — 서버 프록시(가장 안전)**

- 클라이언트에서 직접 Key를 쓰지 않고, 사내 백엔드가 OpenAI 호출을 대행.
- 사용 로그, 요청요금 관리, 입력 필터링(민감정보 차단)을 중앙에서 적용할 수 있음.

## 4. 패키지 설치 및 설정

### A. UPM으로 설치(com.openai.unity)

A.1 Unity 〉 Window 〉 Package Manager
A.2 `+` 〉 *Add package from git URL...* 입력 : 'https://github.com/RageAgainstThePixel/com.openai.unity.git#upm'
A.3 설치 후 'Samples'' 폴더 확인 및 예제 씬으로 실행해보기

### B. OpenUPM 리포지터리 추가

B.1 Edit 〉 Project Settings 〉 Package Manager
B.2 Name: 'OpenUPM', URL: 'https://package.openupm.com', Scope(s): 'com.openai', 'com.utilities'
B.3 Apply 후 Package Manager에서 검색하여 설치

## C. OpenAI-API-dotnet (대안)

### C.1 GitHub에서 Releases의 '.zip' 또는 DLL을 다운로드
### C.2 프로젝트의 'Plugins'나 적절한 위치에 DLL 추가
### C.3 Newtonsoft.Json.dll {ex)net45 등}도 함께 포함(버전에 맞게)

## 5. 기본 채팅 흐름 샘플 (요약)

### 설명: 아래 코드는 Unity UI(TextMeshPro)와 버튼을 사용하여 유저 입력을 받아 OpenAI에 전달하고, 응답을 받아 화면에 표시하는 흐름입니다. (실전에서는 비동기/오류처리/타임아웃/요금제 제약을 추가하세요.)

```
using UnityEngine;
using TMPro;
using UnityEngine.UI;
using OpenAI_API;          // SDK 사용하는 경우
using OpenAI_API.Chat;
using OpenAI_API.Models;
using System.Collections.Generic;
public class OpenAIController : MonoBehaviour {
    public TMP_Text textField;
    public TMP_InputField inputField;
    public Button okBtn;
    private OpenAIAPI api;
    private List<ChatMessage> messages;
    void Start() {
        // 키를 직접 넣지 말고 환경변수/설정에서 읽어오세요
        api                                    =                                    new
OpenAIAPI(System.Environment.GetEnvironmentVariable("OPENAI_API_KEY"));
        StartConversation();
        okBtn.onClick.AddListener(() => GetResponse());
```

```
    }
    void StartConversation() {
        messages = new List<ChatMessage>() {
            new ChatMessage(ChatMessageRole.System, "너는 회사 내부용 서포트 어시스턴트야. 짧고 명료하게 답해줘.")
        };
        textField.text = "Ready";
    }
    private async void GetResponse() {
        if (string.IsNullOrEmpty(inputField.text)) return;
        okBtn.enabled = false;
        var userMessage = new ChatMessage(ChatMessageRole.User, inputField.text);
        messages.Add(userMessage);
        inputField.text = "";
        var chatResult = await api.Chat.CreateChatCompletionAsync(new ChatRequest() {
            Model = Model.ChatGPTTurbo,
            Temperature = 0.1, MaxTokens = 300, Messages = messages
        });
        var response = chatResult.Choices[0].Message.Content;
        messages.Add(new ChatMessage(chatResult.Choices[0].Message.Role, response));
        textField.text = response;
        okBtn.enabled = true;
    }
}
```

**〉 주의: 위 예제는 학습·사내 데모용입니다. 제품 적용 시에는 요청 필터링(민감 정보 제거), 요금 한도 로직, 재시도 및 에러 분기 등을 추가하세요.**

## 6. 한글 폰트(TextMeshPro) 적용

1. 원하는 한글 TTF 파일을 다운로드
2. Unity 〉 Window 〉 TextMeshPro 〉 Font Asset Creator
3. Source Font File에 TTF 넣고 'Generate Font Atlas' 후 저장
4. 코드에서 'TextMeshProUGUI.font = yourGeneratedFontAsset'로 설정

## 7. 보안·비용·테스트 권장사항

- 비용관리: OpenAI 호출은 토큰 기반 비용이 발생. 데모 단계에선 모델(예: GPT-4o vs GPT-4 Turbo)과 MaxTokens를 엄격히 제한하세요.
- 데이터 필터링: 사용자 입력에 개인식별정보(PII)나 민감정보가 포함되지 않도록 프런트/서버에서 마스킹/차단.
- 로깅 정책: 로그에 원본 PII가 남지 않도록 토큰화 또는 해시 처리.
- Rate limit handling: 429 응답을 처리하고 지수 백오프 재시도를 구현.

## 8. 배포 체크리스트

- 키를 클라이언트에 하드코딩하지 않았는가?
- 호출 한도 및 요금 알림(예: 알림/자동 차단)이 설정되었는가?
- 모델 버전(예: gpt-4o, gpt-4-turbo) 변경 시 테스트 케이스가 준비되었는가?
- 로깅·모니터링 및 비용 추적 시스템이 연동되었는가?

## 9. FAQ / 문제 해결

- 한글 깨짐: TextMeshPro 폰트 애셋 미설정.
- JSON 직렬화 오류: Newtonsoft.Json 버전 확인 및 .NET 호환성 설정 확인.
- 패키지 설치 실패: Package Manager의 Git URL 정확성 및 OpenUPM 설정 확인.

## 10) 부록: 샘플 씬 구성 제안

- Chat UI 패널: 입력창, 제출 버튼, 대화 표시 영역
- 설정 패널: 모델 선택, 온도(Temperature) 슬라이더, MaxTokens 입력
- 개발 패널(숨김): API 키 모드(환경변수/에셋), 요청 로그, 호출 카운터