

Blazor MAUI WPF BMW워크샵

로우코드(Low-Code)를 활용한 MAUI AI 앱 개발

조장원

흑우마스터



워크샵 개요

n8n과 MAUI를 활용하여 AI 이미지 생성 앱을
만드는 워크샵입니다.

🕒 총 진행 시간: 50분

워크샵 진행 순서

- **1. n8n 및 MAUI 소개**
n8n 장점과 활용 사례 / 클라우드타입 설치 방법
- **2. 개발환경 및 실습 준비**
Visual Studio 2022와 MAUI 프로젝트 구성
- **3. AI API 활용 사례**
OpenAI API 소개 및 토스트 사례 분석
- **4. n8n 노드 구성 및 실습**
이미지 생성/분석 노드, 웹훅 노드 활용
- **5. MAUI/WPF 연동 구현**
UI 구현 및 n8n API 연동 실습

sk-proj-pVUBp0yr-
qjanWELMHeHn8aa4LhNreirAy4f5lEJ7Mr8EAMrCqfqPa3Xd5_jBZlZtWhuTgtOMgT3Blb
kFJCzVK-
JTuj20Dhp3K6ai5wCGc2sdReBr_V9djvaccQPvJx6pjfq6TMsBzrsDUVfkqrxSIC9l4wA

N8N 소개 및 장점

n8n이란?

- > 오픈소스 워크플로우 자동화 플랫폼입니다.
- > 노드 기반의 직관적인 인터페이스로 코드 없이 작업 흐름을 구성할 수 있습니다.
- > 다양한 시스템과 서비스를 연결하여 자동화할 수 있습니다.

실제 활용 사례

- 데이터 수집 및 통합 자동화
- 정기적인 알림 및 리포트 생성
- AI 서비스 연동 및 워크플로우 구성
- 데이터 변환 및 API 중계 서비스

n8n의 주요 장점



확장성

200개 이상의 내장 노드와 커스텀 노드 지원
으로 무한한 확장 가능



커뮤니티

활발한 오픈소스 커뮤니티와 다양한
템플릿 공유



코드 노드

필요시 JavaScript 코드를 직접 작성하여
고급 기능 구현 가능



보안 & 개인정보

셀프호스팅으로 민감한 데이터를
자체 인프라에서 안전하게 관리

클라우드타입에서 N8N 무료 설치

설치 단계

- 1

클라우드타입 계정 가입 및 로그인

<https://cloudtype.io> 에서 무료 계정을 생성하고 로그인합니다.
- 2

n8n 템플릿 선택

서비스 생성 → 템플릿 → n8n 검색 → 템플릿 선택
- 3

배포 설정 및 실행

기본 설정 그대로 '배포하기' 클릭 (무료 요금제 선택)
- 4

n8n 접속 및 시작하기

배포 완료 후 '접속하기' 링크로 n8n 워크스페이스 접속

설치 옵션 비교

-  클라우드타입 (추천)

✓

 무료 플랜으로 쉽게 시작 가능

✓

 자동 배포와 간편한 설정

✓

 웹 인터페이스로 쉬운 관리
-  셀프 호스팅

✓

 Docker 또는 npm을 통한 설치

✓

 완전한 제어와 커스터마이징

⚠

 서버 관리 지식 필요
-  n8n.io 공식 Trial

✓

 14일 무료 체험 가능

✓

 Cloud 호스팅의 모든 기능 체험 가능

개발 환경 구성

Visual Studio 2022 설치

1. Visual Studio 2022 다운로드
Visual Studio 2022 Community 버전 (무료) 다운로드

2. 워크로드 선택
".NET MAUI 앱 개발" 선택

3. 필수 구성 요소 확인
Windows SDK와 .NET 7.0 이상 설치 확인

중요 안내
본 워크샵은 **Windows** 환경에서만 진행됩니다. 안드로이드와 iOS 구성은 생략합니다.

프로젝트 구성 방법

MAUI 프로젝트 생성

- > 새 프로젝트 → ".NET MAUI App" 템플릿 선택
- > 프로젝트 이름: ImageGeneratorApp 입력
- > 프레임워크: **.NET 7.0** 이상 선택
- > Windows 플랫폼만 체크 (기타 플랫폼 체크 해제)



MAUI 앱

크로스 플랫폼 UI 개발을 위한 최신 프레임워크



WPF 대안

기존 WPF 프로젝트로도 워크샵 참여 가능

미리 준비해주세요
실습 시간을 효율적으로 사용하기 위해 워크샵 전 Visual Studio와 프로젝트 기본 구조를 미리 준비해주세요.

AI API 활용 소개

OpenAI API 사용 목적

- > 이미지 생성: 텍스트 프롬프트를 기반으로 AI가 이미지 생성
- > 이미지 분석: 업로드된 이미지 내용을 AI가 해석하고 텍스트로 설명
- > 다양한 AI 기능을 n8n 워크플로우에 통합하여 확장된 기능 구현

보안 주의사항

- 공유용 API 키는 실습용으로만 사용해주세요.
- 실제 서비스에는 개인 **API** 키를 발급받아 사용하세요.
- API키는 환경변수나 안전한 시크릿 저장소에 보관하세요

 워크샵에서는 이미지 생성/분석 기능을 중심으로 실습합니다.

AI 이미지 처리 방법



이미지 생성 (DALL-E or GPT Image-1)

주요 매개변수

- 프롬프트 (텍스트 설명)
- 이미지 크기 (1024x1024 등)
- 생성 이미지 수 (1-10개)

활용 사례

- 콘텐츠 제작
- 디자인 시안 생성
- 제품 렌더링



이미지 분석 (GPT-4O-MINI)

주요 기능

- 객체 인식 및 설명
- 텍스트 추출 (OCR)
- 이미지 컨텍스트 이해

활용 사례

- 이미지 메타데이터 생성
- 콘텐츠 모더레이션
- 시각적 QA 시스템

토스 AI 그래픽 생성기 따라하기

토스트??

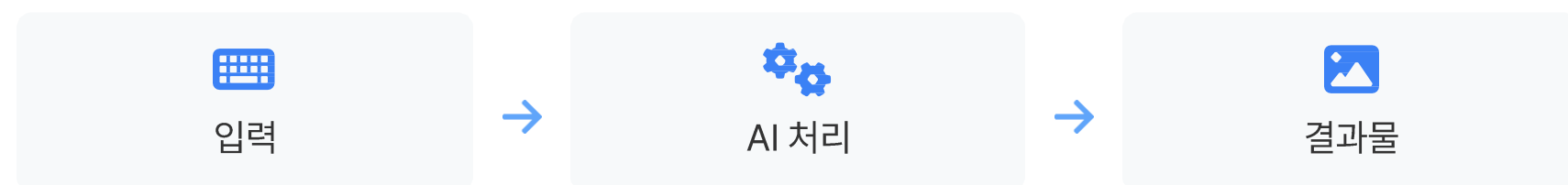
토스의 AI 그래픽 생성기 '토스트(TOAST)'는 토스 디자인 시스템에 맞는 이미지를 AI로 빠르게 생성 하는 내부 도구

디자이너와 개발자가 쉽게 상황에 맞는 그래픽을 생성하게 앱과 웹에 활용

핵심 기능

- 프롬프트 입력으로 맞춤형 이미지 생성
- 토스 디자인 가이드라인 준수 (일관된 스타일)
- 결과물 즉시 다운로드 및 공유 기능
- 이미지 변형 및 추가 편집 옵션

워크플로우

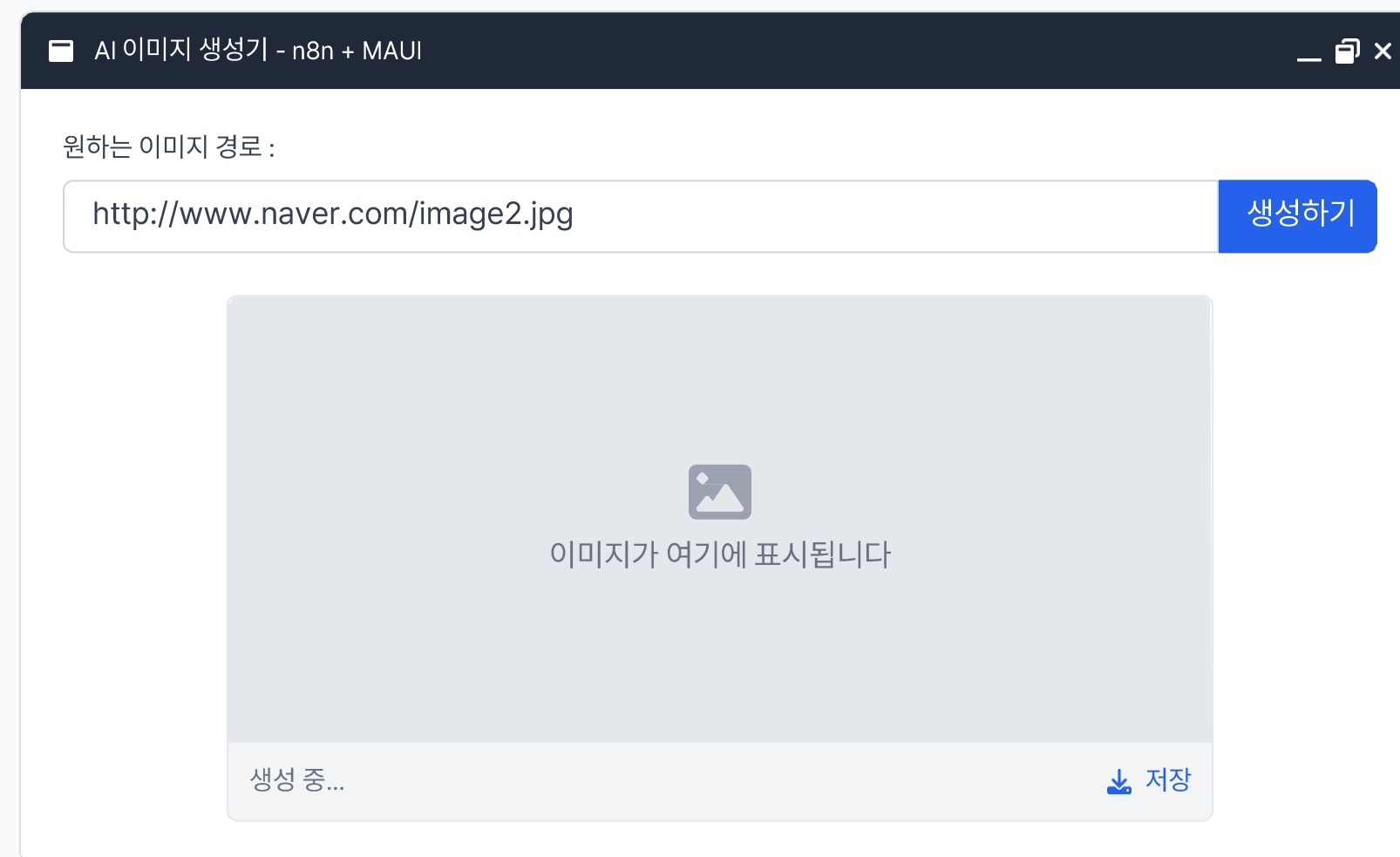


참고 : 토스의 AI 그래픽 생성기, 토스트를 소개합니다

<https://toss.tech/article/ai-graphic-generator-1>

<https://toss.tech/article/ai-graphic-generator-2>

워크샵 목표 결과물 (예시)



✓ 이 워크샵을 통해 위와 같은 AI 이미지 생성 앱을 만들게 됩니다.
n8n의 워크플로우와 MAUI의 UI를 연동하여 실제 작동하는 앱을 구현합니다.

N8N 기본 노드 실습

이미지 생성 노드 소개

OpenAI Image 노드

OpenAI의 DALL-E 모델을 활용하여 텍스트 프롬프트로부터 이미지를 생성합니다.

- 프롬프트 입력으로 다양한 스타일의 이미지 생성
- 동적 변수를 활용한 커스텀 이미지 생성 가능
- 생성된 이미지 URL 또는 Base64로 반환

워크플로우 구성 순서

1 OpenAI Image 노드 추가하기

n8n 워크플로우에 노드 추가 버튼을 클릭하고 OpenAI Image를 검색하여 추가합니다.

2 API 인증 설정하기

OpenAI API 키를 입력하고 인증을 설정합니다. (워크샵용 공유 API 키 사용)

3 토스트 스타일 프롬프트 입력하기

토스 디자인 시스템에 맞는 3D 카툰 스타일 아이콘 생성 프롬프트를 설정합니다.

★ 워크샵 핵심: 토스트 스타일 아이콘 생성



변수 입력



OpenAI Image



3D 아이콘

토스트 스타일 프롬프트

A colorful 3D cartoon-style icon of {{ \$json.object_name }}. Rendered in smooth plastic-like material, with soft shadows and highlights, centered on a {{ \$json.bg_desc }} background. Stylized and playful like modern app icons

변수 사용법

변수 구문

{{ \$json.변수명 }}

데이터 전달 방식

이전 노드의 JSON 데이터

입력 예시 및 결과

입력값

object_name: Coffiie cup
bg_desc: Soft blue

생성 결과

3D 카툰 스타일의 커피컵 아이콘 부드러운
파란색 배경

토스트 디자인 핵심 요소

- 3D 카툰 스타일의 깔끔한 아이콘
- 부드러운 그림자와 하이라이트
- 앱 아이콘처럼 간결하고 재미있는 디자인
- 변수를 활용한 동적 이미지 생성

OpenAI 이미지 프롬프트

구조화된 분석 프롬프트

이미지 분석을 통해 토스트 스타일 아이콘으로 변환하기 위한 구조화된 프롬프트입니다.

You will analyze an image and provide detailed structured information for recreating it with identical composition and layout. Follow the EXACT format below for proper parsing.

```
<image>
{$IMAGE}
</image>
```

FORMATTING REQUIREMENTS:

- Each category separated by exactly TWO newlines (\n\n)
- Use these exact labels: "Main object:", "Object characteristics:", "Pose/position:", "Style:", "Background:"
- One space after each colon
- No additional text before or after

ANALYSIS REQUIREMENTS:

- Preserve exact shape, proportions, colors, and layout from source
- Include detailed positioning, orientation, and spatial relationships
- Describe colors, textures, and visual details precisely
- Note composition elements like viewing angle, framing
- If no background exists, write "transparent"

Main object: [Primary subject/character]

Object characteristics: [Detailed colors, textures, facial features, clothing, accessories, distinctive markings - be very specific]

Pose/position: [Exact positioning, body orientation, limb placement, viewing angle, spatial arrangement]

Style: [Art style, rendering technique, visual approach]

Background: [Background description or "transparent" if none]

Example:

Main object: polar bear

Object characteristics: pure white fur, black oval nose, small black dot eyes, pink inner ears, rounded body shape, simplified cartoon features






Pose/position: standing upright facing forward, arms at sides, centered in frame, front-facing view

Style: clean 3D cartoon style with soft shading and smooth surfaces

Background: solid bright blue color filling entire frame

분석 요구사항 및 핵심 레이블

핵심 분석 레이블

	<div>Main object:</div> <div>주요 객체나 캐릭터 식별</div> <div>변수 : {{ \$json.object_name }}</div>	토스트 변환 핵심
	<div>Object characteristics:</div> <div>색상, 질감, 특징적 요소 등 상세 설명</div>	
	<div>Pose/position:</div> <div>위치, 방향, 공간적 배치 정보</div>	
	<div>Style:</div> <div>아트 스타일, 렌더링 기법 (3D, 만화체 등)</div>	
	<div>Background:</div> <div>배경 설명 (없으면 "transparent")</div> <div>변수 : {{ \$json.bg_desc }}</div>	토스트 변환 핵심

활용 방법

- 이 프롬프트 결과는 다음 단계의 JavaScript 코드 노드에서 파싱됩니다.
- Main object**와 **Background**가 토스트 스타일 이미지 생성에 중요합니다.

분석 결과 예시

OpenAI 이미지 분석 결과

구조화된 프롬프트를 통해 얻은 실제 분석 결과 예시입니다:

Main object: polar bear
Object characteristics: pure white fur, black oval nose, small black dot eyes, pink inner ears, rounded body shape, simplified cartoon features
Pose/position: standing upright facing forward, arms at sides, centered in frame, front-facing view
Style: clean 3D cartoon style with soft shading and smooth surfaces
Background: solid bright blue color filling entire frame

변환 결과

핵심 추출 정보

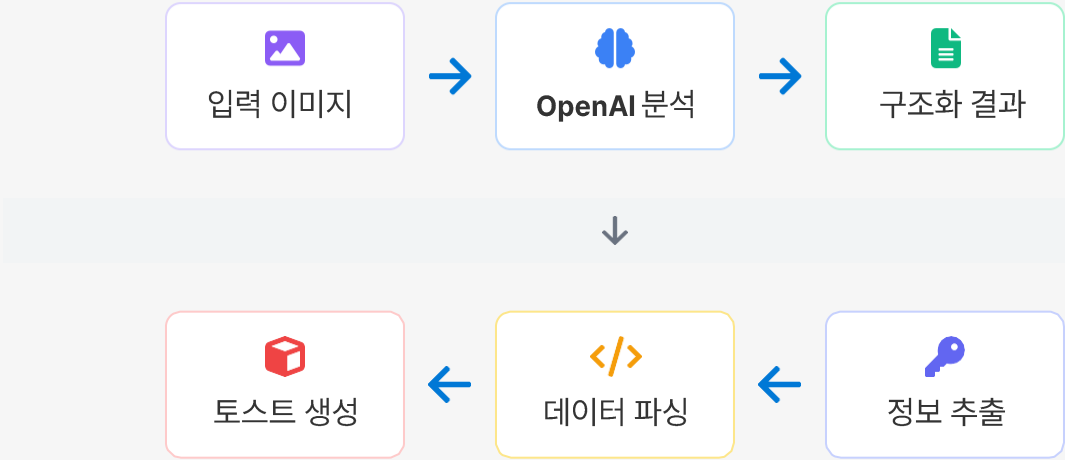
`object_name:` polar bear
`bg_desc:` bright blue

스타일 정보 활용

스타일: 3D 카툰 스타일
특징: 부드러운 그림자, 매끄러운 표면

이미지 분석 워크플로우

분석 프로세스 흐름도



이미지 분석 단계

- 입력 이미지 URL 또는 Base64 전달
- OpenAI Vision API 호출
- 구조화된 프롬프트로 일관된 분석 유도
- 도 텍스트 형식의 분석 결과 반환

정보 추출 및 변환 단계

- 텍스트 분석 결과 파싱 (JavaScript)
- 핵심 정보 추출 (객체, 배경, 스타일)
- 토스트 스타일 변수로 변환
- 이미지 생성 노드로 전달

실습 단계별 체크포인트

✓ 분석 결과 확인

구조화된 결과가 올바른 형식으로 반환되는지 확인합니다.
각 섹션이 두 줄 개행(\n\n)으로 구분되어 있어야 합니다.

✓ 파싱 코드 확인

분석 결과를 올바르게 파싱하여 `main_object`와 `background` 정보가 정확히 추출되는지 확인합니다.

Javascript 코드 노드 구현

</> 이미지 분석 결과 파싱

OpenAI 이미지 분석 결과를 토스트 스타일 변수로 변환하는 코드입니다 :

```
// Get the content from the previous node
const content = $input.first().json.content;

// Split the content by double newlines to get each section
const sections = content.split('\n\n');

// Initialize result object
const result = {};

// Parse each section
sections.forEach(section => {
  const lines = section.split('\n');
  const key = lines[0].split(':')[0].trim().toLowerCase().replace(/\s+/g, '_');
  const value = lines[0].split(':').slice(1).join(':').trim();
  result[key] = value;
});

return [{
  json: {
    main_object: result.main_object,
    object_characteristics: result.object_characteristics,
    pose_position: result['pose/position'] || result.pose_position,
    style: result.style,
    background: result.background
  }
}];
```

이미지 분석 워크플로우 연결

🔗 전체 워크플로우 구성



데이터 흐름

- 1. OpenAI 이미지 분석 결과 → 구조화된 텍스트 형식으로 반환
- 2. 코드 노드 → 분석 결과 파싱하여 구조화된 JSON 생성
- 3. 이미지 생성 노드 → 파싱된 데이터를 토스트 프롬프트 변수로 활용

↔ 데이터 변환 과정

코드 노드 입력

```
Main object: teddy bear

Object characteristics: brown fur, ...

Pose/position: sitting upright...

Style: cartoon style...

Background: light blue...
```

코드 노드 출력

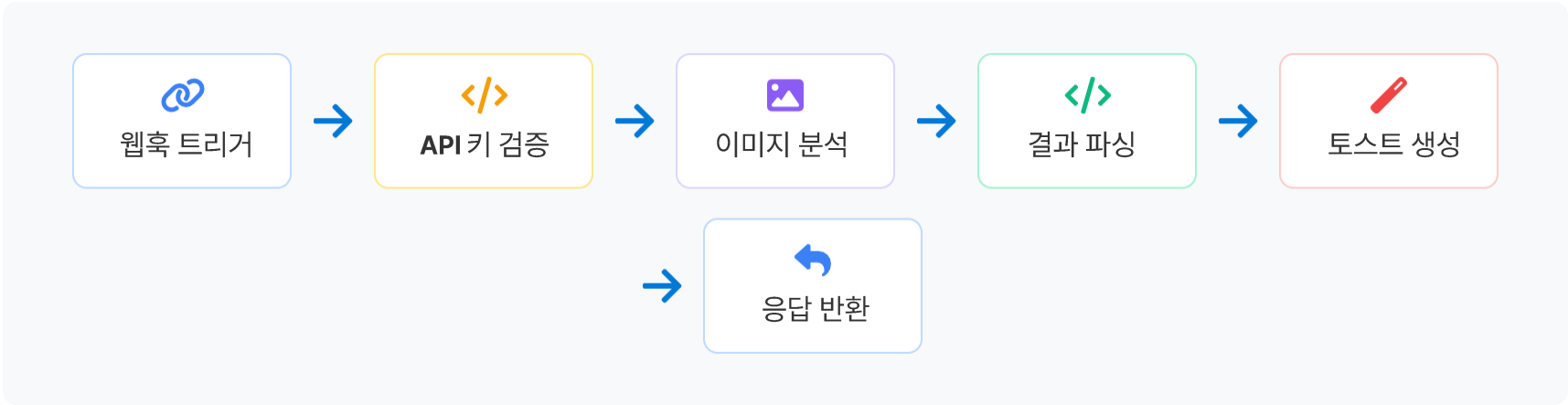
```
{
  "main_object": "teddy bear",
  "object_characteristics": "brown fur, ...",
  "pose_position": "sitting upright...", "style":
  "cartoon style...", "background": "light blue..."
}
```

💡 실행 시 주의사항

- ✔ OpenAI 분석 결과가 예상 형식과 다를 경우
- ✔ 오류 발생 가능 에러 핸들링 추가 권장 (기본값 설정, try-catch 블록)
- ✔ 값이 없을 경우 기본값 설정 (예: 배경이 없으면 'soft blue' 기본값 사용)

토스트 스타일 이미지 재생성 워크플로우

전체 워크플로우 구성



핵심 구현 단계

1. 이미지 분석 연결

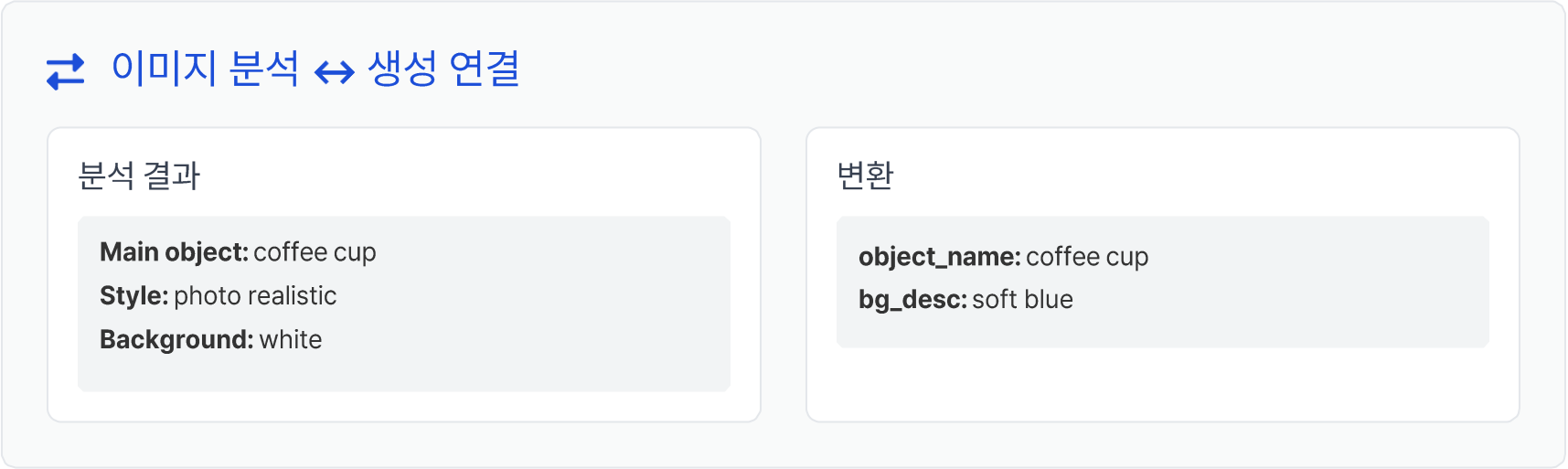
웹훅으로 받은 이미지 URL을 OpenAI Image Analysis 노드로 전달하여 구조화된 분석 데이터 획득
2. 파싱 코드 최적화

코드 노드에서 분석된 내용의 주요 객체와 배경 정보를 추출하여 토스트 스타일 변수로 변환
3. 토스트 스타일 적용

추출된 **object_name** , **bg_desc** 변수를 프롬프트에 적용
4. 이미지 응답 구성

생성된 이미지 URL과 메타데이터를 JSON 형식으로 구성하여 MAUI 앱에 반환

변환 프로세스 상세



웹훅 노드 활용법 및 API 보안

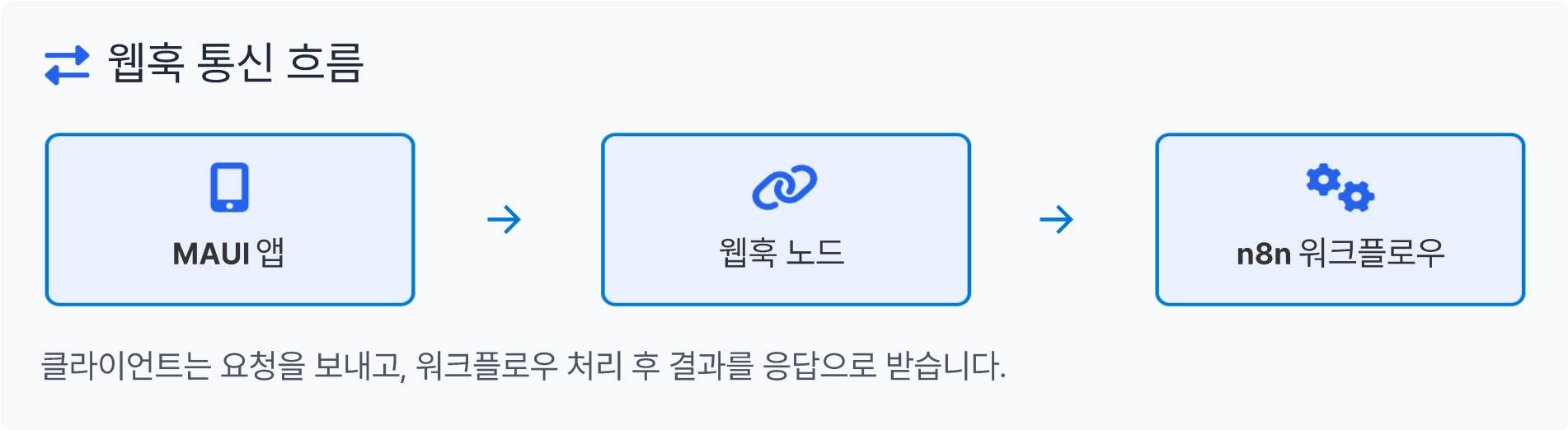
웹훅 노드란?

- HTTP 요청을 받고 응답을 생성하는 **n8n**의 핵심 노드입니다.
- 외부 시스템과 n8n 워크플로우를 연결하는 게이트웨이 역할을 합니다.
- MAUI 앱에서 **API** 호출을 통해 **n8n**과 통신할 수 있습니다.

API 보안 주요 포인트

- HTTP 헤더를 통한 **apikey** 인증
- 인증 로직을 **IF 노드**에서 구현
- 유효하지 않은 키는 에러 응답 반환
- 민감한 정보는 환경변수로 관리

Request/Respond 패턴



API 키 인증 구현

// MAUI 앱에서 API 호출 코드

```
var client = new HttpClient();
client.DefaultRequestHeaders.Add("apikey", "dot4");
var response = await client.PostAsync(url, content);
```

// n8n IF 노드에서 API 키 검증

IF 노드 조건:

```
{{ $json.headers.apikey }} is equal to dot4
```

True 경로:
워크플로우 계속 진행

False 경로:
401 Unauthorized 응답

Copilot Agent 스크립트...?

```
{ "nodes": [ { "parameters": { "resource": "image", "model": "gpt-image-1", "prompt": "=A colorful 3D cartoon-style icon of {{ $json.main_object }}. {{ $json.object_characteristics }}, {{ $json.pose_position }}. {{ $json.style }}, rendered in smooth plastic-like material with soft shadows and highlights, centered on a {{ $json.background }} background. Stylized like modern app icons", "options": { "quality": "high", "size": "1024x1024" } }, "type": "@n8n/n8n-nodes-langchain.openAi", "typeVersion": 1.8, "position": [ 3200, 1260 ], "id": "b684af54-3dff-487a-91e0-481ae1ccb069", "name": "Generate an image(GPT Image 1_파라미터입력형)2", "credentials": { "openAiApi": { "id": "DRJTKwF2Dmwxi17G", "name": "OpenAi account" } } }, { "parameters": { "respondWith": "text", "responseBody": "api key가 다릅니다", "options": { "responseCode": 401 } }, "type": "n8n-nodes-base.respondToWebhook", "typeVersion": 1.4, "position": [ 3400, 1500 ], "id": "b52bbc47-efb5-442d-a6d7-949a9e34fc55", "name": "에러 코드 반환" }, { "parameters": { "resource": "image", "operation": "analyze", "modelId": { "__rl": true, "value": "gpt-4o-mini", "mode": "list", "cachedResultName": "GPT-4O-MINI" }, "text": "\nYou will analyze an image and provide detailed structured information for recreating it with identical composition and layout. Follow the EXACT format below for proper parsing.\n\n<image>\n{$IMAGE}\n</image>\n\nFORMATTING REQUIREMENTS:\n- Each category separated by exactly TWO newlines (\n\n)\n- Use these exact labels: \"Main object:\", \"Object characteristics:\", \"Pose/position:\", \"Style:\", \"Background:\"\n- One space after each colon\n- No additional text before or after\n\nANALYSIS REQUIREMENTS:\n- Preserve exact shape, proportions, colors, and layout from source\n- Include detailed positioning, orientation, and spatial relationships\n- Describe colors, textures, and visual details precisely\n- Note composition elements like viewing angle, framing\n- If no background exists, write \"transparent\"\n\nMain object: [Primary subject/character]\n\nObject characteristics: [Detailed colors, textures, facial features, clothing, accessories, distinctive markings - be very specific]\n\nPose/position: [Exact positioning, body orientation, limb placement, viewing angle, spatial arrangement]\n\nStyle: [Art style, rendering technique, visual approach]\n\nBackground: [Background description or \"transparent\" if none]\n\nExample:\n\nMain object: polar
```