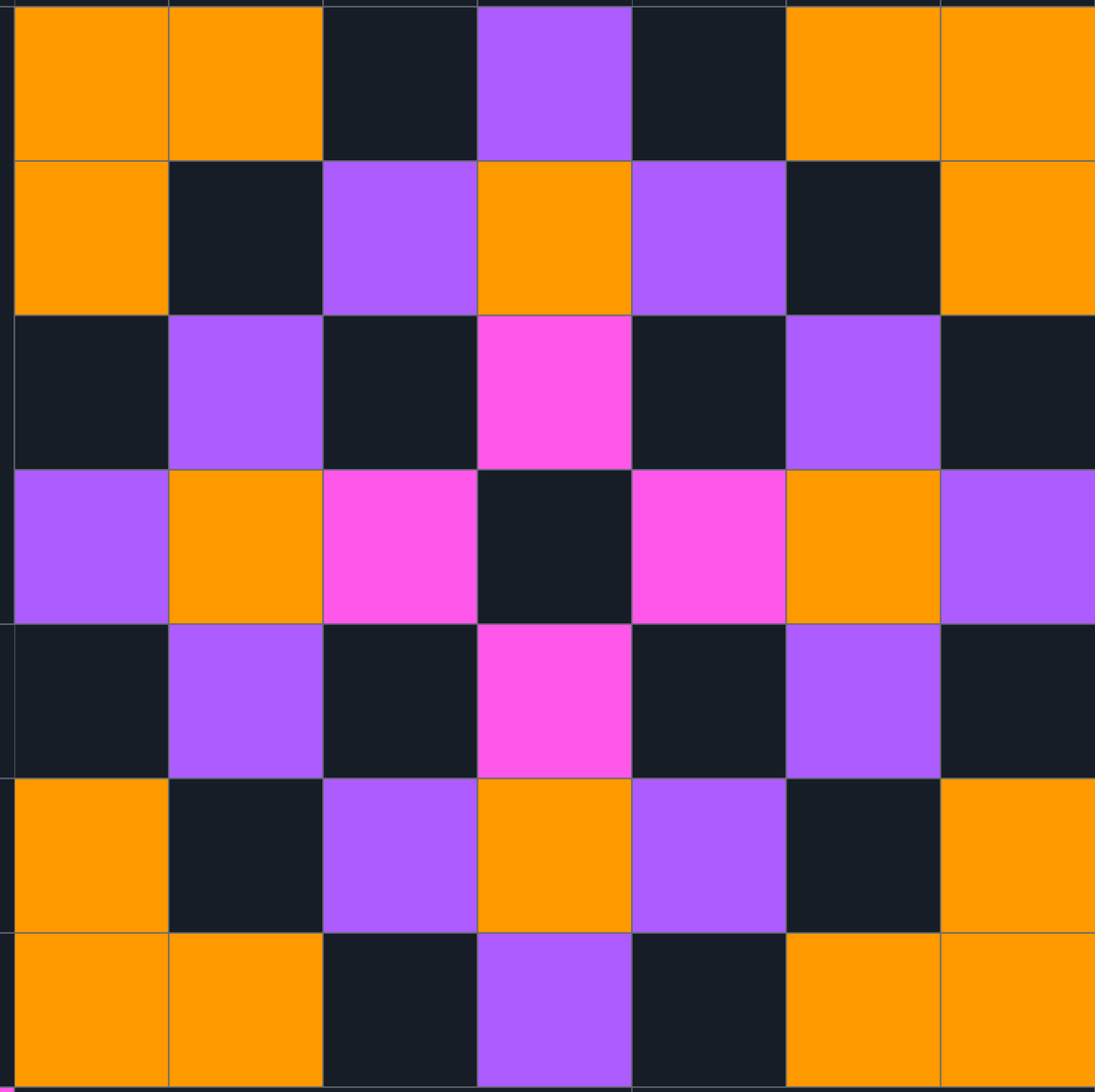
ASBG SESSION 03

Serverless Architecture

맥도날드 주문 시스템으로 이해하는 서버리스

장지원 · 2026. 11. 03



Serverless에도 서버는 있다

줄어드는 것은 서버가 아니라

서버 인프라를 직접 운영하는 부담입니다.

AWS가 맡는 일

실행 환경 준비 · 용량 확장 · 인프라 운영

개발자가 맡는 일

코드 · 데이터 · 권한 · 오류 처리



프로비저닝 · 서비스에 필요한 자원을 준비하고 할당하는 과정

01 · SERVERLESS

주방 운영을 AWS에 맡긴다면?

직접 운영

EC2 + Auto Scaling

인스턴스와 OS를 관리
확장 정책과 인스턴스 수를 구성
지속 실행과 세밀한 제어에 적합

실행 인프라 위임

AWS Lambda

함수 코드와 설정을 배포
실행 환경 준비와 확장은 AWS가 관리
이벤트 기반 작업에 적합

개발자의 시간이 인프라 운영에서 주문 처리 로직으로 이동합니다.

EC2 + Auto Scaling과 AWS Lambda

구분	EC2 + Auto Scaling	AWS Lambda
실행 단위	가상 서버 인스턴스	함수 실행 환경
관리 범위	OS와 인스턴스 운영	AWS가 실행 인프라 관리
확장 방식	인스턴스 수 조정	요청에 맞춰 동시 실행 증가
비용 기준	주로 인스턴스 실행 시간	주로 호출 수와 실행 자원
잘 맞는 작업	지속 실행, 세밀한 환경 제어	간헐적이거나 변동이 큰 작업

두 방식 모두 자동 확장이 가능합니다. 선택 기준은 워크로드입니다.

02 · ORDER SYSTEM

Lambda는 주문이 들어온 걸 어떻게 알까?

Lambda가 계속 지켜보는 것은 아니다

다른 서비스가 전달한 이벤트나 호출이
함수 실행의 출발점이 됩니다.

즉, 주문이 들어오면 직원이 스스로 알아채는 것이 아니라
주문 창구가 직원을 호출합니다.

오늘 사용할 실행 계기

HTTP 요청
파일 업로드
데이터 변경

```
키오스크 버튼 클릭
→ API Gateway 요청 수신
→ Lambda 호출
```

이벤트는 시스템에서 발생한 사건 또는 작업을 유발하는 입력입니다.

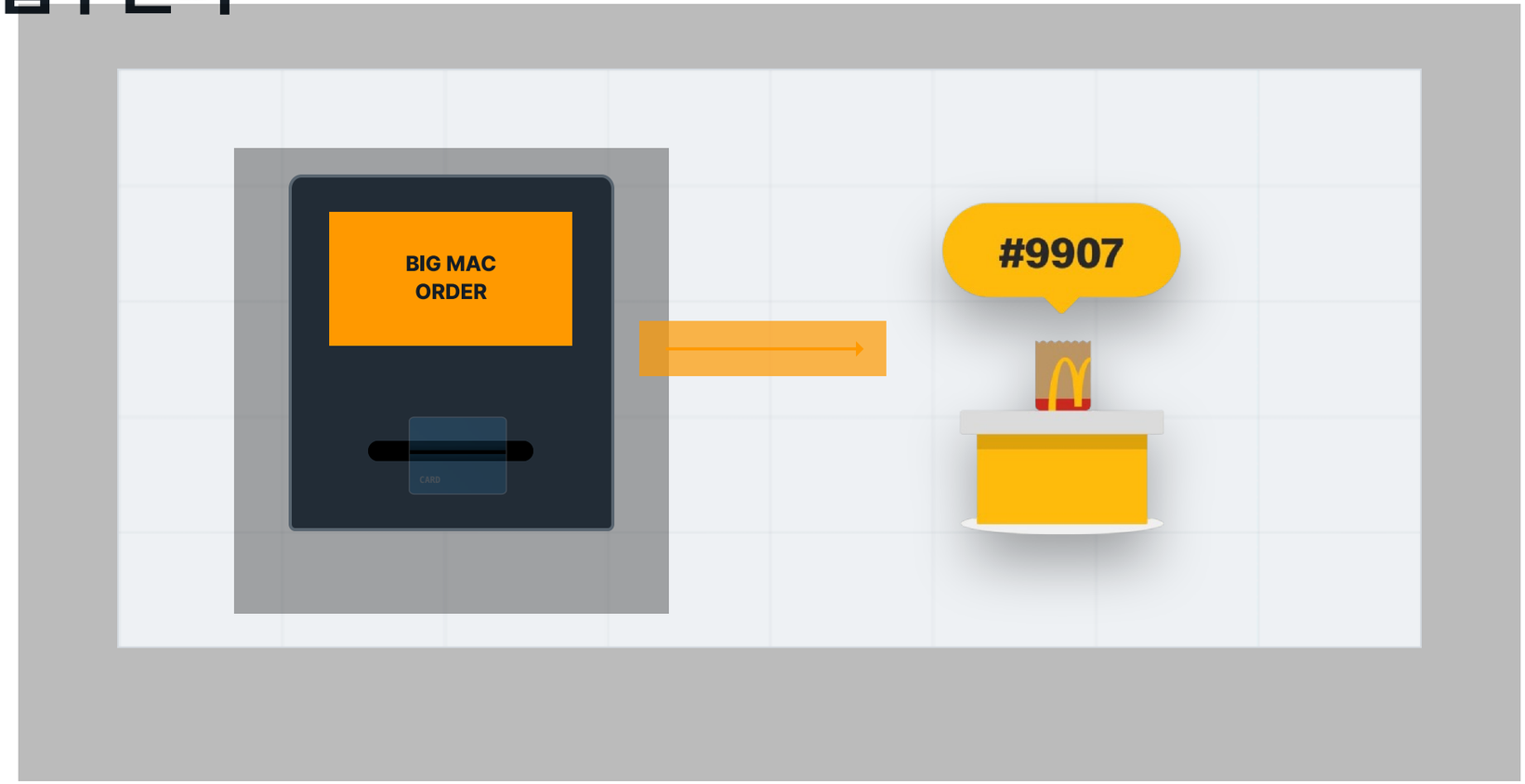
02 · ORDER SYSTEM · 01

키오스크가 주문과 결제를 접수한다

손님은 주방 구조를 몰라도
정해진 주문 창구만 이용하면 됩니다.

카드를 넣어 결제하면 API Gateway가
POST /orders 요청을 받고,
주문 번호를 만들어 다음 처리 단계로 전달합니다.

> 키오스크 = API Gateway
> 요청의 입구



결제가 완료되면 주문 #9907이 접수되고 Lambda 호출로 이어집니다.

02 · ORDER SYSTEM · 02

직원이 주문표를 받아 검증하고 처리한다

접수된 주문은
AWS Lambda Handler의 입력이 됩니다.

직원은 주문표를 받아 필수값을 확인하고,
주문 정보를 저장한 뒤 빅맥을 만드는
비즈니스 로직을 순서대로 실행합니다.

> **직원 = AWS Lambda**
> **검증과 로직 실행**



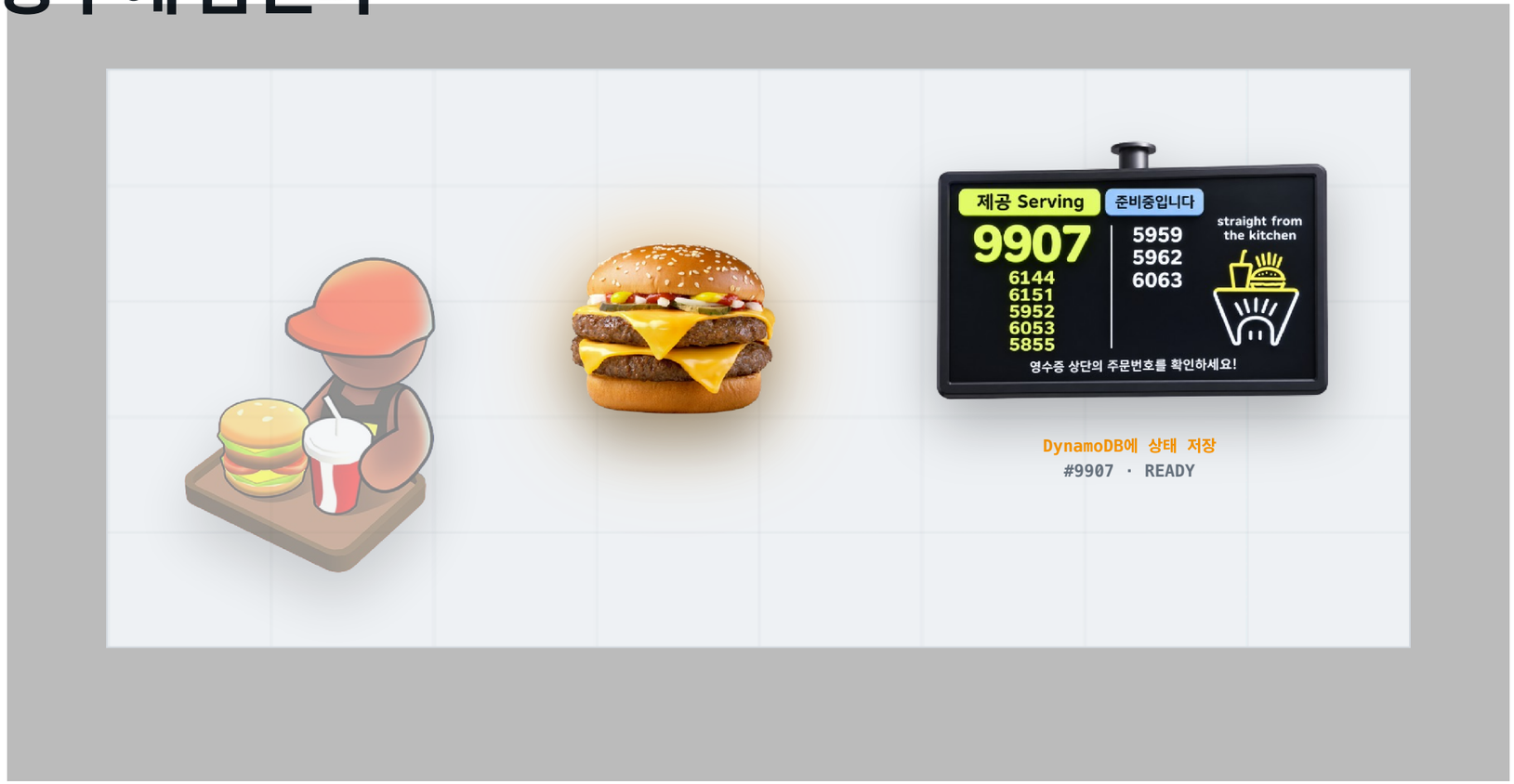
주문 접수 → 입력값 검증 → 주문 정보 저장 → 비즈니스 로직 실행

빅맥을 만들고 결과를 주문 장부에 남긴다

처리가 끝났다는 사실은
직원의 기억이 아니라 장부에 남겨야 합니다.

빅맥이 완성되면 주문 번호와 처리 상태를
DynamoDB에 저장합니다.
다음 요청도 같은 장부에서 결과를 조회할 수 있습니다.

> **주문 장부 = DynamoDB**
> **지속 상태 저장**



DynamoDB에 상태 저장
#9907 · READY

DynamoDB에 저장된 #9907 상태를 주문 현황판에 표시합니다.

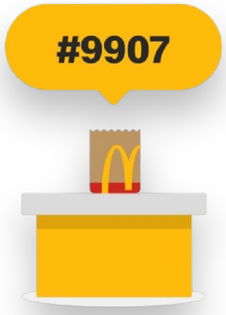
02 · ORDER SYSTEM · COMPLETE

한 건의 주문이 지나가는 전체 경로



손님

Client
주문 고객



주문 접수

API Gateway
요청의 입구



검증과 처리

AWS Lambda
로직 실행



결과 생성

Business Logic
빅맥 완성



상태 저장

DynamoDB
status = READY

모든 작업이 끝나면 Lambda가 HTTP 201 응답을 반환합니다. 사용자는 같은 주문 번호로 처리 결과를 확인합니다.

직원이 바뀌면 이전 주문을 기억할까?

```
total_orders = 0

def lambda_handler(event, context):
    global total_orders
    total_orders += 1
    return total_orders
```

실행 환경 메모리 · 다음 호출까지 유지된다고 보장할 수 없음

DynamoDB · 다음 호출에도 필요한 상태를 지속 저장

03 · CONSTRAINTS

신입 직원에게 금고 비밀번호까지?

필요한 권한

dynamodb:GetItem

dynamodb:PutItem

dynamodb:UpdateItem

불필요한 권한

테이블 삭제

다른 데이터 조회

관리자 권한

최소 권한 원칙 · 업무에 필요한 권한만 부여

03 · CONSTRAINTS

함수가 계속 실행되면 어떻게 될까?

0초

실행 시작

요청을 받고 처리를 시작

6초

정상 범위

실습 지연은 아직 10초 이내

10초

Timeout

오늘 실습에서 정한 최대 실행 시간

초과

실행 중단

직전까지 끝난 작업은 코드가 책임

Warm Start는 중단된 함수를 이어 실행하는 기능이 아닙니다.

04 · SYNCHRONOUS PROCESSING

햄버거가 나올 때까지 기다리는 동기 처리

1

주문 요청

손님이 키오스크에서 주문

2

주문 접수

API Gateway가 요청 전달

3

빅맥 조리

Lambda가 실행되는 동안 손님은 대기

4

HTTP 201

상태 저장 후 완료 응답

동시성은 함께 처리하는 양, 동기·비동기는 결과를 기다리는 방식의 문제입니다.

04 · SYNCHRONOUS PROCESSING

빅맥 조리가 1초에서 6초로 늘어난다면?

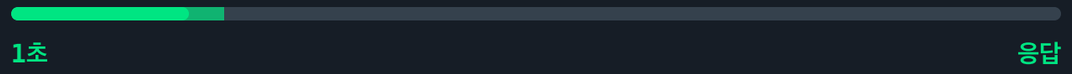
Case A · 조리 지연 1초

주문 접수

빅맥 조리

상태 저장

HTTP 201



Case B · 조리 지연 6초

주문 접수

빅맥 조리

상태 저장

HTTP 201



조리 시간이 길어지면 동기 요청의 응답도 함께 늦어집니다.

빅맥이 완성될 때까지 기다려야 할까?

주문은 먼저 접수하고,
조리는 나중에 해도 되지 않을까?

Next Session · Asynchronous Architecture