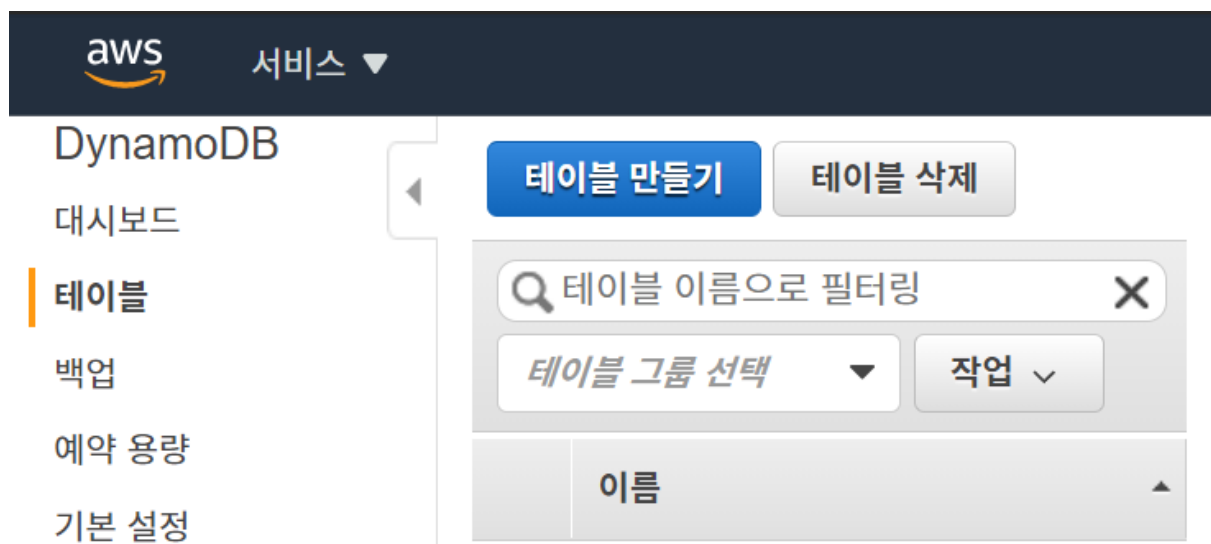




Lambda Chat, Dashboard, DynamoDB, MBTI 인수인계

DynamoDB

1. 테이블 만들기 클릭



2. 정보 입력

DynamoDB 테이블 만들기

[자습서](#)

DynamoDB는 스키마가 없는 데이터베이스로서, 테이블 이름과 기본 키만 필요로 합니다. 테이블의 기본 키는 각 파티션에서 항목을 고유하게 식별하고, 데이터를 분할하며, 데이터를 정렬하는 한 개 또는 여러 개의 속성으로 구성됩니다.

테이블 이름*



기본 키* 파티션 키

문자열

☐ 정렬 키 추가

테이블 설정

기본 설정을 사용하면 테이블을 가장 빠르게 시작할 수 있습니다. 이러한 기본 설정은 지금 또는 테이블을 만든 후에 수정할 수 있습니다.

☒ 기본 설정 사용

- 보조 인덱스 없음
- Auto Scaling 용량은 목표 사용률 70%, 최소 용량 읽기 5개 및 쓰기 5개로 설정
- 유효 데이터를 기본 암호화 유형으로 암호화합니다.

[+ 태그 추가](#) **신규 기능!**

CloudWatch 또는 SNS(Simple Notification Service)에 대한 AWS 프리 티어를 초과할 경우 추가 요금이 적용될 수 있습니다. 고급 알람 설정은 CloudWatch 관리 콘솔에서 사용할 수 있습니다.

[취소](#)[생성](#)

만들어야 할 테이블

Aa 테이블 이름	≡ 기본 키	≡ 추가 키
<u>chat_connections</u>	connectionId (문자열)	
<u>chat_log</u>	roomId (문자열)	requestTimeEpoch (번호)
<u>MBTI-product</u>	id (번호)	

RDS

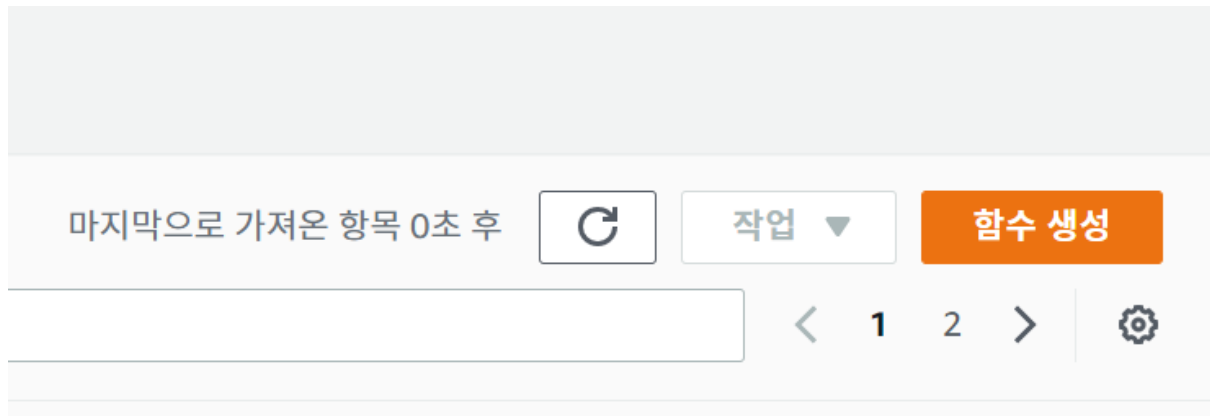
`exec/Lambda/LanguageGenerator` 의 `(1)RDS만들기및설정하기.pdf` 를 참고하여 만든다.

MySQL Workbench 같은 프로그램에서 `make_table.sql`, `mbti_relationship.sql`, `procedure.sql` 을 순서대로 실행한다.

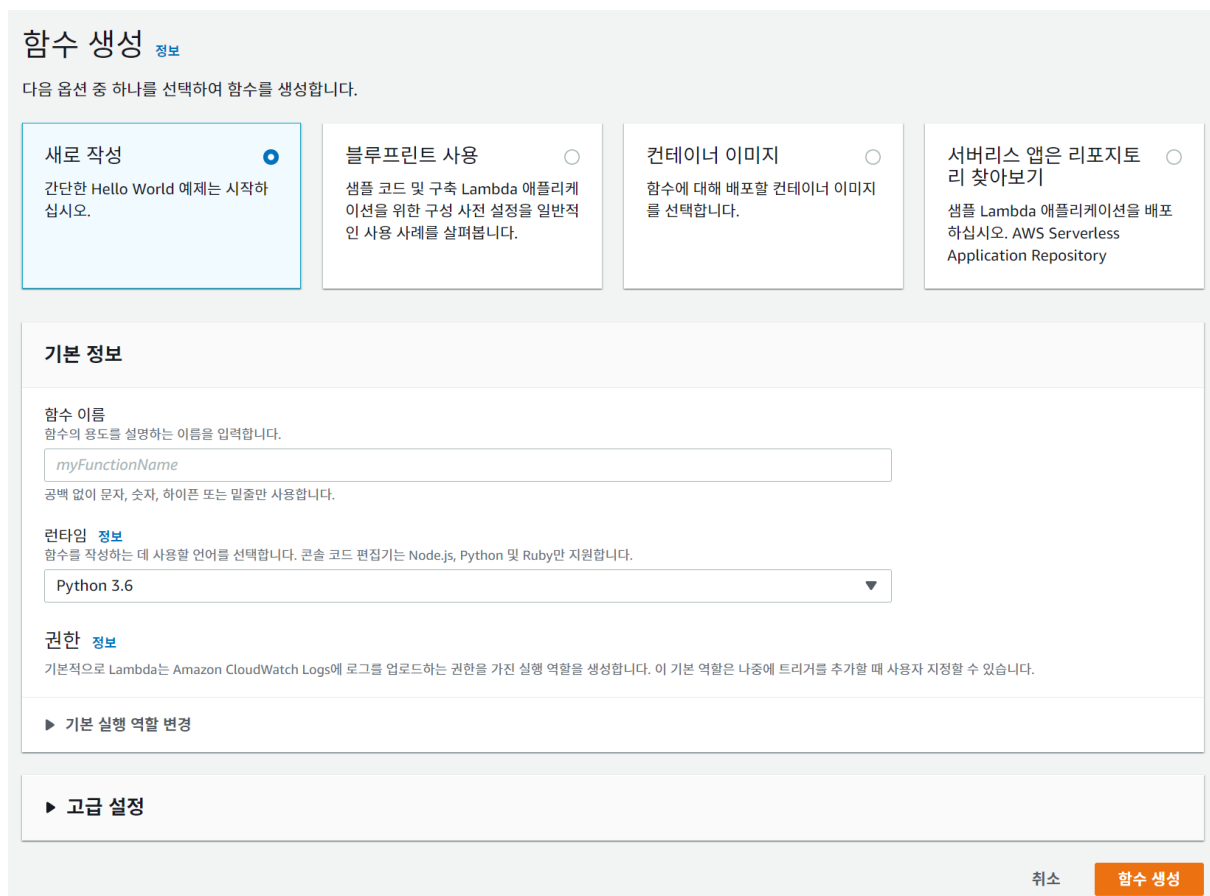
AWS Lambda

만드는법

1. 함수 생성 클릭



2. 함수 이름, 런타임 설정 후 함수 생성 클릭

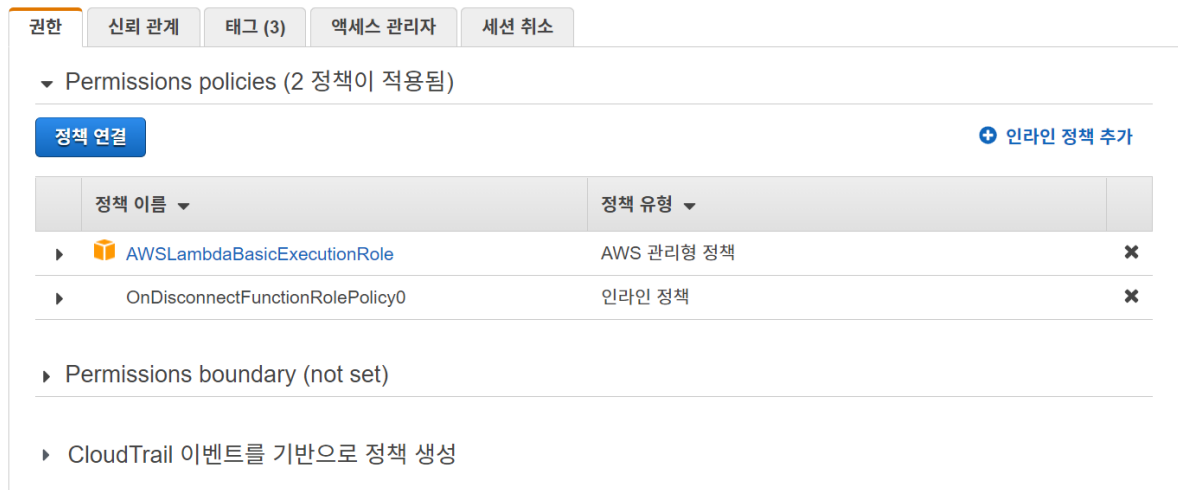


IAM 등록 방법

1. 구성 → 권한 → 역할 이름 클릭



2. 인라인 정책 추가 클릭

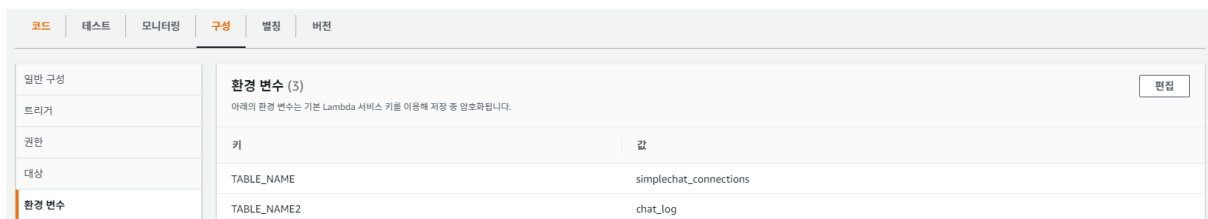


3. JSON 클릭 후 IAM 추가



환경 변수 등록 방법

1. 구성 → 환경 변수 → 편집 클릭(환경 변수 필요 시)



2. 환경 변수 키 / 값 작성 후 저장

환경 변수

환경 변수를 함수 코드에서 액세스할 수 있는 키-값 페어로 정의할 수 있습니다. 이렇게 하면 함수 코드를 변경하지 않고도 구성 설정을 저장하는 데 유용합니다. [자세히 알아보기](#)

키	값	
TABLE_NAME	simplechat_connections	제거
TABLE_NAME2	chat_log	제거
TABLE_NAME3	chat_log2	제거

환경 변수 추가

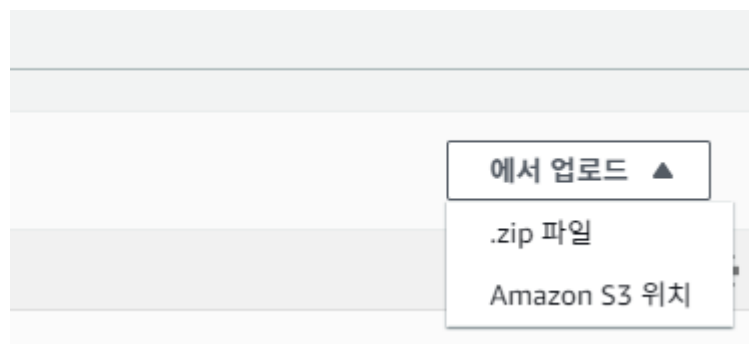
▶ 암호화 구성

취소

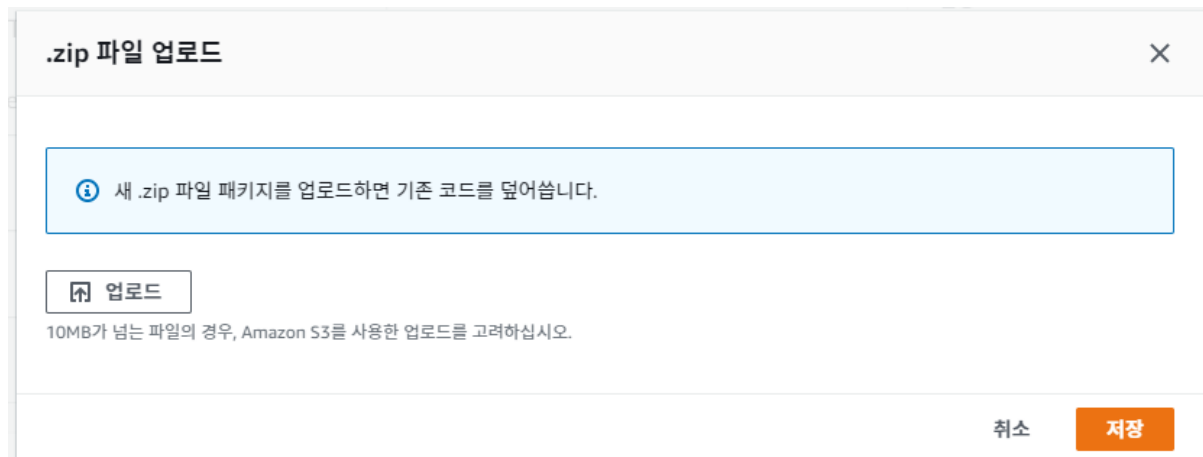
저장

zip 파일로 업로드 방법

1. zip 파일에서 업로드 클릭



2. 업로드할 zip 파일 찾아서 등록 후 저장



MBTI 함수

1. MBTI_landing_page.zip (zip 파일로 업로드하기)

- 런타임: Python 3.6
- requirements: pymysql
- MBTI 성격 유형 테스트의 메인 페이지에서 각 성격 유형의 검사 결과 비율과 총 검사 결과 count를 반환하는 함수
- line 6 의 host 에 AWS RDS 엔드포인트 주소, user 에 유저명, password 에 비밀번호, db 에 스키마명 넣기

2. MBTI_result.zip (zip 파일로 업로드하기)

- 런타임: Python 3.6
- requirements: pymysql
- MBTI 성격 유형을 input으로 받으면 해당하는 유형의 정보를 output으로 반환하는 함수
- line 7 의 host 에 AWS RDS 엔드포인트 주소, user 에 유저명, password 에 비밀번호, db 에 스키마명 넣기

3. MBTI_update_count.zip (zip 파일로 업로드하기)

- 런타임: Python 3.6
- requirements: pymysql
- MBTI 성격 유형 테스트의 마지막 질문에 대한 응답을 하고 나면 데이터베이스에 해당하는 유형의 count를 +1 하는 함수
- line 8 의 host 에 AWS RDS 엔드포인트 주소, user 에 유저명, password 에 비밀번호, db 에 스키마명 넣기

4. MBTI_randProduct.py

- 런타임: Python 3.6
- 삼성전자 제품 중 다섯 개를 랜덤으로 추출해서 반환하는 함수
- IAM

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "dynamodb:BatchGetItem",
        "dynamodb:BatchWriteItem",
        "dynamodb:ConditionCheckItem",
        "dynamodb:PutItem",
        "dynamodb:DescribeTable",
        "dynamodb>DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:Scan",
        "dynamodb:Query",
        "dynamodb:UpdateItem"
      ],
      "Resource": "*"
    }
  ]
}
```

DynamoDB 함수

1. DynamoDB_upload_product.zip (zip 파일로 업로드하기)

- 런타임: Python 3.8
- requirements: boto3
- DynamoDB에 삼성 제품을 무작위 순서대로 입력하는 함수. 압축파일 내의 json 파일의 데이터를 db에 입력한다.
- IAM

```
"Version": "2012-10-17",
"Statement": [
  {
    "Sid": "VisualEditor0",
    "Effect": "Allow",
    "Action": [
      "dynamodb:BatchGetItem",
```



```

        "dynamodb:BatchWriteItem",
        "dynamodb:ConditionCheckItem",
        "dynamodb:PutItem",
        "dynamodb:DescribeTable",
        "dynamodb>DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:Scan",
        "dynamodb:Query",
        "dynamodb:UpdateItem"
    ],
    "Resource": "*"
}
]
}

```

Chat 함수

1. Chat_load_prevChat.zip (zip 파일로 업로드하기)

- 런타임: Python 3.6
- requirements: boto3
- MBTI 성격 유형 별 결과 페이지에서 채팅 버튼을 눌렀을 때 이전 채팅 기록을 불러 오는 함수
- IAM

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "dynamodb:BatchGetItem",
        "dynamodb:BatchWriteItem",
        "dynamodb:ConditionCheckItem",
        "dynamodb:PutItem",
        "dynamodb:DescribeTable",
        "dynamodb>DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:Scan",
        "dynamodb:Query",
        "dynamodb:UpdateItem"
      ],
      "Resource": "*"
    }
  ]
}

```

```
]
}
```

2. Chat_on_connect.js

- 런타임: Node.js 12.x
- 환경변수
 - `TABLE_NAME` : `chat_connections`
 - `TABLE_NAME2` : `chat_log`
- 채팅을 위해 웹소켓 연결을 시작하는 함수
- IAM

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "dynamodb:BatchGetItem",
        "dynamodb:BatchWriteItem",
        "dynamodb:ConditionCheckItem",
        "dynamodb:PutItem",
        "dynamodb:DescribeTable",
        "dynamodb>DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:Scan",
        "dynamodb:Query",
        "dynamodb:UpdateItem"
      ],
      "Resource": "*"
    }
  ]
}
```

3. Chat_enter_room.js

- 런타임: Node.js 12.x
- 환경변수
 - `TABLE_NAME` : `chat_connections`
- MBTI 성격 유형 별 채팅방에 입장하는 함수
- IAM

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "dynamodb:BatchGetItem",
        "dynamodb:BatchWriteItem",
        "dynamodb:ConditionCheckItem",
        "dynamodb:PutItem",
        "dynamodb:DescribeTable",
        "dynamodb>DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:Scan",
        "dynamodb:Query",
        "dynamodb:UpdateItem"
      ],
      "Resource": "*"
    }
  ]
}
```

4. Chat_send_message.js

- 런타임: Node.js 12.x
- 환경변수
 - `TABLE_NAME` : `chat_connections`
- 채팅방에 메시지를 전달하는 함수
- IAM

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "dynamodb:BatchGetItem",
        "dynamodb:BatchWriteItem",
        "dynamodb:ConditionCheckItem",
        "dynamodb:PutItem",
        "dynamodb:DescribeTable",
        "dynamodb>DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:Scan",
        "dynamodb:Query",
        "dynamodb:UpdateItem"
      ],

```

```

        "Resource": "*"
    }
]
}

```

5. Chat_on_disconnect.js

- 런타임: Node.js 12.x
- 환경변수
 - `TABLE_NAME` : `chat_connections`
- 연결이 끊어지면 데이터베이스에서 `connectionId`를 삭제하는 함수
- IAM

```

{
  "Statement": [
    {
      "Action": [
        "dynamodb:GetItem",
        "dynamodb:DeleteItem",
        "dynamodb:PutItem",
        "dynamodb:Scan",
        "dynamodb:Query",
        "dynamodb:UpdateItem",
        "dynamodb:BatchWriteItem",
        "dynamodb:BatchGetItem",
        "dynamodb:DescribeTable",
        "dynamodb:ConditionCheckItem"
      ],
      "Resource": [
        "*"
      ],
      "Effect": "Allow"
    }
  ]
}

```

Dashboard

1. Dashboard_CW_API.zip (zip 파일로 업로드하기)

- 런타임: Python 3.6
- requirements: boto3

- Dashboard에 보여줄 lambda 함수 관련 cloudwatch log 결과를 이미지를 반환하는 함수

API Gateway

MBTI API

1. HTTP API 구축

API 유형 선택

HTTP API

OIDC 및 OAuth2와 같은 기능과 기본 CORS 지원이 내장된, 지연 시간이 짧고 비용 효율적인 REST API를 구축합니다.

다음과 호환됩니다.

Lambda, HTTP 백엔드

가져오기

구축

2. 경로 생성

14

Cross-Origin Resource Sharing

CORS 구성

[구성](#)[지우기](#)

CORS를 사용하면 브라우저에서 다른 도메인의 리소스를 로드할 수 있습니다. API에 대해 CORS를 구성하면 API Gateway가 백엔드 통합에서 반환된 CORS 헤더를 무시합니다. 자세한 내용은 [CORS 설명서](#)를 참조하십시오.

Access-Control-Allow-Origin



Access-Control-Allow-Headers

헤더가 허용되지 않음

Access-Control-Allow-Methods

메서드가 허용되지 않음

=Access-Control-Expose-Headers

노출 헤더가 허용되지 않음

Access-Control-Max-Age

0 초

Access-Control-Allow-Credentials

☐ 아니요

Chat API

1. WebSocket API 구축

WebSocket API

채팅 애플리케이션 또는 대시보드와 같은 실시간 사용 사례를 위해 지속적 연결을 사용하여 WebSocket API를 구축합니다.

다음과 호환됩니다.

Lambda, HTTP, AWS 서비스

[구축](#)

2. 각 경로에 lambda function 연결

경로

작업 ▾

경로 선택 표현식 ⓘ

`$request.body.action`



새 경로 키

`$connect`

`$disconnect`

`enterroom`

`sendmessage`



`$default`

websocket 경로 - lambda

Aa websocket 경로	≡ lambda
<u><code>\$connect</code></u>	Chat_on_connect
<u><code>\$disconnect</code></u>	Chat_on_disconnect
<u><code>enterroom</code></u>	Chat_enter_room
<u><code>sendmessage</code></u>	Chat_send_message

3. API 배포 (안 하면 변경 사항 적용이 안 됨)

경로

작업 ▾

경로 선택 표현식 ⓘ
\$request.body.action

API 작업

API 배포

API 삭제