Add a permission의

AWS IoT TwinMaker

AWS IoT TwinMaker: Add a permission의

Table of Contents

무엇입니까 AWS IoT TwinMaker?

AWS IoT TwinMaker 물리적 시스템과 디지털 시스템의 운영 디지털 트윈을 구축하는 데 사용할 수 있는 AWS IoT 서비스입니다. AWS IoT TwinMaker 다양한 실제 센서, 카메라 및 엔터프라이즈 애플리케이션의 측정 및 분석을 사용하여 디지털 시각화를 생성하여 실제 공장, 건물 또는 산업 플랜트를 추적하는 데 도움이 됩니다. 이 실제 데이터를 사용하여 작업을 모니터링하고 오류를 진단 및 해결하며 작업을 최적화할 수 있습니다.

디지털 트윈은 시스템과 시스템의 모든 물리적 및 디지털 구성 요소를 실시간으로 디지털로 표현한 것입니다. 이는 시스템의 실제 구조, 상태 및 동작을 모방하도록 데이터로 동적으로 업데이트됩니다. 이를 사용하여 비즈니스 성과를 창출할 수 있습니다.

최종 사용자는 사용자 인터페이스 애플리케이션을 사용하여 디지털 트윈의 데이터와 상호 작용합니다.

작동 방식

디지털 트윈을 만들기 위한 최소 요구 사항을 충족하려면 다음을 수행해야 합니다.

- 물리적 위치에서 장치, 장비, 공간 및 프로세스를 모델링하십시오.
- 이러한 모델을 센서 데이터 카메라 피드와 같은 중요한 컨텍스트 정보를 저장하는 데이터 소스에 연결하십시오.
- 사용자가 데이터와 통찰력을 이해하는 데 도움이 되는 시각화를 작성하여 비즈니스 의사결정을 보다 효율적으로 수행하십시오.
- 최종 사용자가 디지털 트윈을 사용하여 비즈니스 성과를 높일 수 있도록 하십시오.

AWS IoT TwinMaker 다음과 같은 기능을 제공하여 이러한 문제를 해결합니다.

- 개체 구성 요소 시스템 지식 그래프: 지식 그래프에서 장치, 장비, 공간 및 프로세스를 모델링하기 위한 도구를 AWS IoT TwinMaker 제공합니다.

  이 지식 그래프에는 시스템에 대한 메타데이터가 포함되어 있으며 다양한 위치에 있는 데이터에 연결할 수 있습니다. AWS IoT TwinMaker AWS IoT SiteWise 및 Kinesis Video Streams에 저장된 데이터를 위한 내장 커넥터를 제공합니다. 다른 위치에 저장된 데이터에 대한 사용자 지정 커넥터를 만들 수도 있습니다.

지식 그래프와 커넥터는 서로 다른 위치에 있는 데이터를 쿼리하기 위한 단일 인터페이스를 제공합니다.

- 씬 컴포저: AWS IoT TwinMaker 콘솔은 3D로 씬을 생성할 수 있는 씬 구성 도구를 제공합니다. 이전에 구축한 3D/CAD 모델을 업로드하여 웹 디스플레이에 최적화하고.gltf 또는.glb 형식으로 변환합니다. 그런 다음 장면 구성기를 사용하여 여러 모델을 단일 장면에 배치하여 해당 작업을 시각적으로 표현할 수 있습니다.

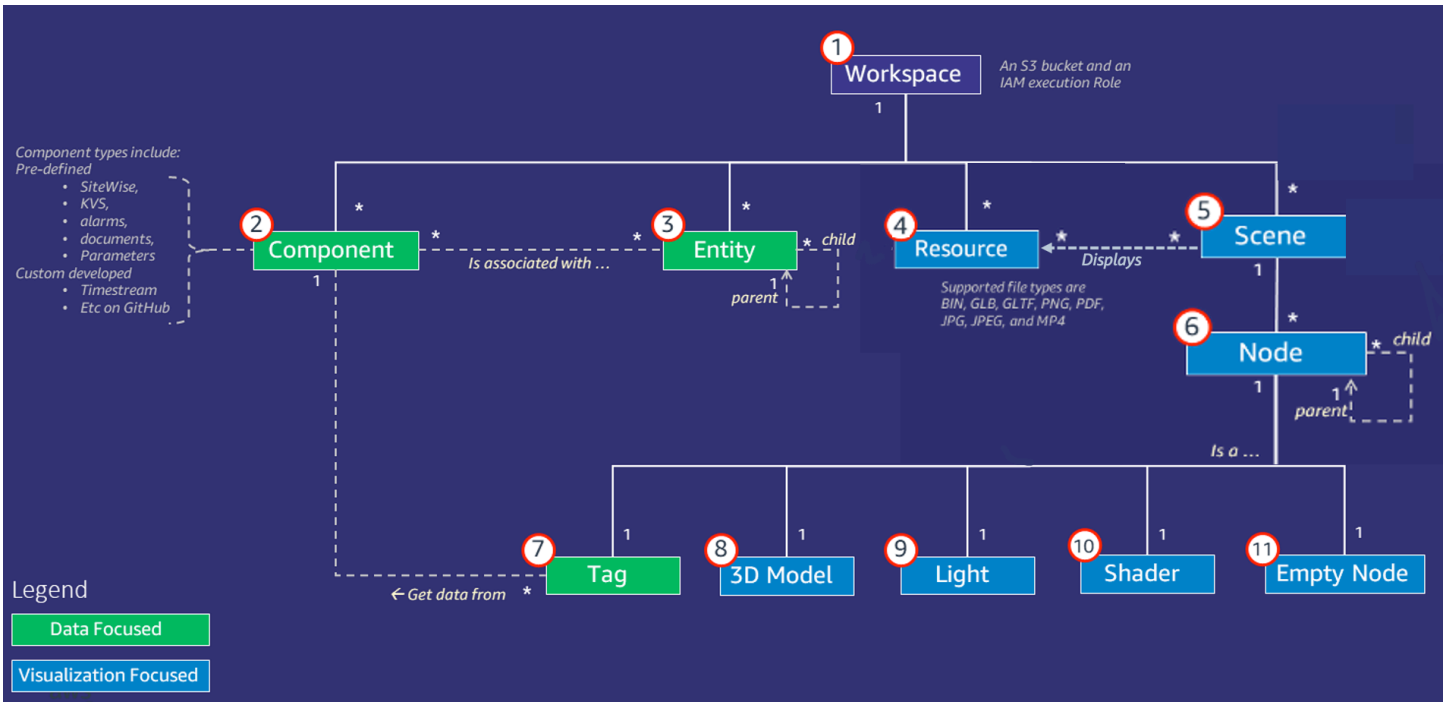
  장면에 데이터를 오버레이할 수도 있습니다. 예를 들어 장면 위치에 태그를 만들어 센서의 온도 데이터에 연결할 수 있습니다. 이렇게 하면 데이터가 위치와 연결됩니다.

- 애플리케이션: 최종 AWS IoT TwinMaker 사용자를 위한 대시보드 애플리케이션을 구축하는 데 사용할 수 있는 Grafana 및 Amazon Managed Grafana용 플러그인을 제공합니다.

- 타사 도구: Mendix는 산업용 IoT를 위한 완벽한 솔루션을 제공하기 AWS IoT TwinMaker 위해 협력합니다. Mendix를 사용한 린 일일 관리 애플리케이션 워크숍을 참조하고 AWS IoT TwinMaker Kinesis AWS IoT TwinMaker Video Streams 등과 같은 AWS 서비스와 함께 Mendix 로우 코드 애플리케이션 개발 플랫폼 (LCAP) 사용을 시작하는 방법을 알아보십시오. AWS IoT SiteWise

핵심 개념 및 구성 요소

다음 다이어그램은 의 주요 개념이 어떻게 조화를 이루는지 보여줍니다. AWS IoT TwinMaker



> **Note**
>
> 다이어그램의 별표 (*) 는 관계를 나타냅니다. one-to-many 각 관계에 대한 할당량은 AWS IoT TwinMaker 엔드포인트 및 할당량을 참조하십시오.

다음 단원에서는 다이어그램에 표시된 개념을 설명합니다.

워크스페이스

워크스페이스는 디지털 트윈 애플리케이션을 위한 최상위 컨테이너입니다. 이 워크스페이스 내에 디지털 트윈을 위한 개체, 구성 요소, 장면 자산 및 기타 리소스의 논리적 세트를 생성합니다. 이는 또한 디지털 트윈 애플리케이션과 여기에 포함된 리소스에 대한 액세스를 관리하는 보안 경계 역할도 합니다. 각 작업 영역은 작업 영역 데이터가 저장되는 Amazon S3 버킷에 연결됩니다. IAM 역할을 사용하여 작업 공간에 대한 액세스를 제한할 수 있습니다.

워크스페이스는 여러 구성 요소, 개체, 장면, 리소스를 포함할 수 있습니다. 구성 요소 유형, 개체, 장면 또는 리소스는 하나의 워크스페이스 내에서만 존재합니다.

개체-구성 요소 모델

AWS IoT TwinMaker 지식 그래프를 사용하여 시스템을 모델링하는 데 사용하는 도구를 제공합니다. entity-component-based 개체 구성 요소 아키텍처를 사용하여 물리적 시스템을 표현할 수 있습니다. 이 개체 구성 요소 모델은 개체, 구성 요소 및 관계로 구성됩니다. 개체 구성 요소 시스템에 대한 자세한 내용은 개체 구성 요소 시스템을 참조하십시오.

개체

개체는 해당 요소의 기능을 캡처하는 디지털 트윈의 요소를 디지털로 표현한 것입니다. 이 요소는 물리적 장비, 개념 또는 프로세스일 수 있습니다. 개체에는 개체와 관련된 구성 요소가 있습니다. 이러한 구성 요소는 관련 개체에 대한 데이터 및 컨텍스트를 제공합니다.

를 사용하면 엔티티를 사용자 지정 계층 구조로 구성하여 보다 효율적으로 관리할 수 있습니다. AWS IoT TwinMaker개체 및 구성 요소 시스템의 기본 뷰는 계층적입니다.

구성 요소

구성 요소는 장면의 개체에 대한 컨텍스트와 데이터를 제공합니다. 개체에 구성 요소를 추가합니다. 구성 요소의 수명은 개체의 수명과 관련이 있습니다.

구성 요소는 문서 목록 또는 지리적 위치의 좌표와 같은 정적 데이터를 추가할 수 있습니다. 또한 시계열 데이터가 포함된 시스템 (예: 다른 시계열 클라우드 히스토리언) 을 비롯한 다른 시스템에 연결하는 함수를 가질 수도 있습니다. AWS IoT SiteWise

구성 요소는 데이터 소스와 AWS IoT TwinMaker간의 연결을 설명하는 JSON 문서에 의해 정의됩니다. 구성 요소는 외부 데이터 소스 또는 기본 제공되는 데이터 소스를 설명할 수 있습니다. AWS IoT TwinMaker구성 요소는 JSON 문서에 지정된 Lambda 함수를 사용하여 외부 데이터 소스에 액세스합니다. 워크스페이스에는 많은 구성 요소가 포함될 수 있습니다. 구성 요소는 관련 개체를 통해 태그에 데이터를 제공합니다.

AWS IoT TwinMaker 콘솔에서 추가할 수 있는 몇 가지 기본 제공 구성 요소를 제공합니다. TimeStream Telemetry 및 Geospatial 좌표와 같은 데이터 소스에 연결할 사용자 지정 구성 요소를 직접 만들 수도 있습니다. 이러한 예로는 TimeStream 텔레메트리, 지리공간 구성 요소, Snowflake와 같은 타사 데이터 소스에 대한 커넥터 등이 있습니다.

AWS IoT TwinMaker 일반적인 사용 사례를 위해 다음과 같은 유형의 기본 제공 구성 요소를 제공합니다.

- 문서 (예: 지정된 URL에 있는 사용자 설명서 또는 이미지)
- 시계열 (예: AWS IoT SiteWise의 센서 데이터).
- 알람 (예: 외부 데이터 소스의 시계열 알람)
- 동영상 (Kinesis Video Streams에 연결된 IP 카메라)
- 추가 데이터 소스에 연결하기 위한 사용자 지정 구성 요소. 예를 들어, AWS IoT TwinMaker 개체를 외부에 저장된 시계열 데이터에 연결하는 사용자 지정 커넥터를 만들 수 있습니다.

데이터 소스

데이터 소스는 디지털 트윈의 소스 데이터 위치입니다. AWS IoT TwinMaker 두 가지 유형의 데이터 소스를 지원합니다.

- 계층 커넥터를 사용하면 외부 모델을 AWS IoT TwinMaker로 지속적으로 동기화할 수 있습니다.
- 시계열 커넥터: AWS IoT SiteWise등의 시계열 데이터베이스에 연결할 수 있습니다.

속성

속성은 구성 요소에 포함된 정적 및 시계열 기반 값입니다. 구성 요소를 개체에 추가하면 구성 요소의 속성이 개체의 현재 상태에 대한 세부 정보를 설명합니다.

AWS IoT TwinMaker 다음과 같은 세 가지 속성을 지원합니다.

- 단일 값, non-time-series 속성 - 이러한 속성은 일반적으로 정적 키-값 쌍이며 관련 엔티티의 메타데이터와 AWS IoT TwinMaker 함께 직접 저장됩니다.
- 시계열 속성 - 이러한 AWS IoT TwinMaker 속성의 시계열 저장소에 대한 참조를 저장합니다. 기본값은 최신값입니다.
- 관계 속성 — 이러한 속성은 다른 개체 또는 구성 요소에 대한 참조를 저장합니다. 예를 들어 `seen_by`은(는) 카메라 개체를 해당 카메라에 의해 직접 시각화되는 다른 개체와 연결할 수 있는 관계 구성 요소입니다.

통합 데이터 쿼리 인터페이스를 사용하여 이기종 데이터 소스에서 속성 값을 쿼리할 수 있습니다.

시각화

디지털 트윈의 3차원 표현을 보강한 다음 Grafana에서 보는 AWS IoT TwinMaker 데 사용합니다. 장면을 만들려면 기존 CAD 또는 기타 3D 파일 유형을 사용하십시오. 그런 다음 데이터 오버레이를 사용하여 디지털 트윈에 관련 데이터를 추가합니다.

장면

장면은 연결된 데이터에 대한 시각적 컨텍스트를 제공하는 3차원 표현입니다. AWS IoT TwinMaker장면은 전체 환경에 대해 단일 gltf (GL 전송 형식) 또는 glb 3D 모델을 사용하거나 여러 모델로 구성된 구성을 사용하여 만들 수 있습니다. 장면에는 장면의 관심 지점을 나타내는 태그도 포함됩니다.

씬은 시각화를 위한 최상위 컨테이너입니다. 장면은 하나 이상의 노드로 구성됩니다.

워크스페이스에는 여러 장면이 포함될 수 있습니다. 예를 들어 워크스페이스에는 시설의 각 층마다 하나의 장면이 포함될 수 있습니다.

리소스

씬에는 리소스가 표시되며, 리소스는 콘솔에 AWS IoT TwinMaker 노드로 표시됩니다. 장면에는 많은 리소스가 포함될 수 있습니다.

리소스는 장면을 만드는 데 사용되는 이미지 및 `glTF` 기반 3차원 모델입니다. 리소스는 단일 장비 또는 전체 사이트를 나타낼 수 있습니다.

.gltf 또는.glb 파일을 워크스페이스 리소스 라이브러리에 업로드한 다음 장면에 추가하여 리소스를 장면에 배치합니다.

증강 사용자 인터페이스

AWS IoT TwinMaker 를 사용하면 센서 데이터와 같은 중요한 컨텍스트와 정보를 씬의 위치에 추가하는 데이터 오버레이로 씬을 보강할 수 있습니다.

노드: 노드는 태그, 조명, 3차원 모델의 인스턴스입니다. 장면 계층 구조에 구조를 추가하기 위해 비워 둘 수도 있습니다. 예를 들어, 여러 노드를 하나의 빈 노드 아래에 그룹화할 수 있습니다.

태그: 태그는 (개체를 통해) 구성 요소의 데이터를 나타내는 노드 유형입니다. 태그는 구성 요소 하나에만 연결할 수 있습니다. 태그는 장면의 특정 `x,y,z` 좌표 위치에 추가되는 주석입니다. 태그는 개체 속성을 사용하여 이 장면 부분을 지식 그래프에 연결합니다. 태그를 사용하여 장면에 있는 항목의 동작이나 시각적 모양(예: 알람)을 구성할 수 있습니다.

조명: 장면에 조명을 추가하여 특정 오브젝트에 초점을 맞추거나 그림자를 드리워 물리적 위치를 표시할 수 있습니다.

3차원 모델: 3차원 모델은 리소스로 가져온 .gltf 또는.glb 파일을 시각적으로 표현한 것입니다.

> **Note**
>
> AWS IoT TwinMaker 심각한 신체 상해 또는 사망으로 이어지거나 환경 또는 재산 피해를 야기할 수 있는 위험한 환경 또는 중요 시스템의 작동 시 또는 이와 연계하여 사용해서는 안 됩니다. 사용을 통해 수집된 데이터는 사용 사례에 적합한 정확성을 AWS IoT TwinMaker 평가해야 합니다. AWS IoT TwinMaker 물리적 시스템이 안전하게 작동하고 있는지 평가하기 위한 목적으로 물리적 시스템을 사람이 모니터링하는 대신 사용해서는 안 됩니다.

시작하기 AWS IoT TwinMaker

이 섹션의 항목에서는 다음을 수행하는 방법에 대해 설명합니다.

- 새 워크스페이스를 생성하고 설정합니다.
- 개체를 생성하고 개체에 구성 요소를 추가합니다.

사전 조건:

첫 번째 워크스페이스와 장면을 생성하려면 다음 AWS 리소스가 필요합니다.

- AWS 계정.
- 에 대한 IAM 서비스 역할입니다 AWS IoT TwinMaker. 이 역할은 AWS IoT TwinMaker 콘솔에서 새 AWS IoT TwinMaker 워크스페이스를 생성할 때 기본적으로 자동으로 생성됩니다.

  에서 새 IAM 서비스 역할을 AWS IoT TwinMaker 자동으로 생성하도록 선택하지 않은 경우 이미 생성한 역할을 지정해야 합니다.

  이 서비스 역할을 생성 및 관리하는 방법에 대한 지침은 ???을(를) 참조하십시오.

  IAM 서비스 역할에 대한 자세한 내용은 AWS 서비스에 권한을 위임할 역할 생성을 참조하십시오.

  > **Important**
  >
  > 이 서비스 역할에는 서비스가 Amazon S3 버킷을 읽고 쓸 수 있는 권한을 부여하는 연결된 정책이 있어야 합니다.는이 역할을 AWS IoT TwinMaker 사용하여 사용자를 대신하여 다른 서비스에 액세스합니다. 또한 서비스가 역할을 수임할 수 AWS IoT TwinMaker 있도록이 역할과 간에 신뢰 관계를 할당해야 합니다. 트윈이 다른 AWS 서비스와 상호 작용하는 경우 해당 서비스에 필요한 권한도 추가합니다.

주제

- 에 가입 AWS 계정
- 에 대한 서비스 역할 생성 및 관리 AWS IoT TwinMaker
- 워크스페이스 생성
- 첫 번째 개체 생성
- AWS 계정 설정

에 가입 AWS 계정

를 시작하려면이 AWS필요합니다 AWS 계정. 생성에 대한 자세한 AWS 계정내용은 AWS Account Management 참조 안내서의 시작하기 AWS 계정를 참조하세요.

에 대한 서비스 역할 생성 및 관리 AWS IoT TwinMaker

AWS IoT TwinMaker 은(는) 서비스 역할을 사용하여 사용자 대신 다른 서비스의 리소스에 액세스할 수 있도록 해야 합니다. 이 역할은와 신뢰 관계가 있어야 합니다 AWS IoT TwinMaker. 워크스페이스를 생성할 때 이 역할을 워크스페이스에 할당해야 합니다. 이 주제에는 일반적인 시나리오에 대한 권한을 구성하는 방법을 보여 주는 정책 예제가 포함되어 있습니다.

신뢰 부여

다음 정책은 역할과 간의 신뢰 관계를 설정합니다 AWS IoT TwinMaker. 이 신뢰 관계를 워크스페이스에 사용하는 역할에 할당하십시오.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "iottwinmaker.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Amazon S3 권한

다음 정책은 역할이 Amazon S3 버킷에서 읽고 삭제하고 쓸 수 있는 권한을 부여합니다. 워크스페이스는 Amazon S3에 리소스를 저장하므로 모든 워크스페이스에 Amazon S3 권한이 필요합니다.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucket*",
        "s3:GetObject",
        "s3:ListBucket",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:DeleteObject"
      ],
      "Resource": [
        "arn:aws:s3:::*/DO_NOT_DELETE_WORKSPACE_*"
      ]
    }
  ]
}
```

> **Note**
>
> 워크스페이스를 생성할 때는 워크스페이스에서 사용 중임을 나타내는 파일을 Amazon S3 버킷에 AWS IoT TwinMaker 생성합니다. 이 정책은 워크스페이스를 삭제할 때 해당 파일을 삭제할 수 있는 AWS IoT TwinMaker 권한을 부여합니다.
> AWS IoT TwinMaker 는 워크스페이스와 관련된 다른 객체를 배치합니다. 워크스페이스를 삭제할 때 이러한 오브젝트를 삭제하는 것은 사용자의 책임입니다.

특정 Amazon S3 버킷에 대한 권한 할당

AWS IoT TwinMaker 콘솔에서 워크스페이스를 생성할 때 Amazon S3 버킷을 생성하도록 AWS IoT TwinMaker 선택할 수 있습니다. 다음 AWS CLI 명령을 사용하여이 버킷에 대한 정보를 찾을 수 있습니다.

```
aws iottwinmaker get-workspace --workspace-id workspace name
```

다음 예제에서는 이 명령의 출력 형식을 보여줍니다.

```
{
    "arn": "arn:aws:iottwinmaker:region:account Id:workspace/workspace name",
    "creationDateTime": "2021-11-30T11:30:00.000000-08:00",
    "description": "",
    "role": "arn:aws:iam::account Id:role/service role name",
    "s3Location": "arn:aws:s3:::bucket name",
    "updateDateTime": "2021-11-30T11:30:00.000000-08:00",
    "workspaceId": "workspace name"
}
```

특정 Amazon S3 버킷에 대한 권한을 할당하도록 정책을 업데이트하려면 ## ## 값을 사용하십시오.

다음 정책은 사용자의 역할이 특정 Amazon S3 버킷에서 읽고 삭제하고 쓸 수 있도록 허용합니다.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucket*",
        "s3:GetObject",
        "s3:ListBucket",
        "s3:PutObject"
```

```
      ],
      "Resource": [
        "arn:aws:s3:::bucket name",
        "arn:aws:s3:::bucket name/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:DeleteObject"
      ],
      "Resource": [
        "arn:aws:s3:::iottwinmakerbucket/DO_NOT_DELETE_WORKSPACE_*"
      ]
    }
  ]
}
```

기본 제공 커넥터에 대한 권한

워크스페이스가 내장 커넥터를 사용하여 다른 AWS 서비스와 상호 작용하는 경우이 정책에 해당 서비스에 대한 권한을 포함해야 합니다. com.amazon.iotsitewise.connector 구성 요소 유형을 사용하는 경우 AWS IoT SiteWise에 대한 권한을 포함해야 합니다. 구성 요소 유형에 대한 자세한 정보는 ???을(를) 참조하세요.

> **Note**
>
> 사용자 지정 구성 요소 유형을 사용하여 다른 AWS 서비스와 상호 작용하는 경우 구성 요소 유형에서 함수를 구현하는 Lambda 함수를 실행할 수 있는 권한을 역할에 부여해야 합니다. 자세한 내용은 ??? 단원을 참조하십시오.

다음 예제에서는 AWS IoT SiteWise 정책에를 포함하는 방법을 보여줍니다.

JSON

```
{
  "Version":"2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucket*",
        "s3:GetObject",
        "s3:ListBucket",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::bucket name",
        "arn:aws:s3:::bucket name/*"
      ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "iotsitewise:DescribeAsset"
        ],
        "Resource": "arn:aws:s3:::bucket name"
        },
    {
        "Effect": "Allow",
        "Action": [
            "iotsitewise:DescribeAssetModel"
        ],
        "Resource": "arn:aws:s3:::bucket name"
        },
    {
      "Effect": "Allow",
      "Action": [
        "s3:DeleteObject"
      ],
      "Resource": [
        "arn:aws:s3:::*/DO_NOT_DELETE_WORKSPACE_*"
      ]
    }
  ]
}
```

com.amazon.iotsitewise.connector 구성 요소 유형을 사용하고 속성 데이터를 읽어야 하는 경우 정책에 다음 권한을 포함해야 AWS IoT SiteWise합니다.

```
...
{
    "Action": [
        "iotsitewise:GetPropertyValueHistory",
    ],
    "Resource": [
        "AWS IoT SiteWise asset resource ARN"
    ],
    "Effect": "Allow"
},
...
```

com.amazon.iotsitewise.connector 구성 요소 유형을 사용하고 속성 데이터를에 기록해야 하는 경우 정책에 다음 권한을 포함해야 AWS IoT SiteWise합니다.

```
...
{
    "Action": [
        "iotsitewise:BatchPutPropertyValues",
    ],
    "Resource": [
        "AWS IoT SiteWise asset resource ARN"
    ],
    "Effect": "Allow"
},
...
```

com.amazon.iotsitewise.connector.edgevideo 구성 요소 유형을 사용하는 경우 AWS IoT SiteWise 및 Kinesis Video Streams에 대한 권한을 포함해야 합니다. 다음 예제 정책은 정책에 AWS IoT SiteWise 및 Kinesis Video Streams 권한을 포함하는 방법을 보여줍니다.

```
...
{
    "Action": [
```

```
            "iotsitewise:DescribeAsset",
            "iotsitewise:GetAssetPropertyValue"
        ],
        "Resource": [
            "AWS IoT SiteWise asset resource ARN for the Edge Connector for Kinesis Video
 Streams"
        ],
        "Effect": "Allow"
    },
    {
        "Action": [
            "iotsitewise:DescribeAssetModel"
        ],
        "Resource": [
            "AWS IoT SiteWise model resource ARN for the Edge Connector for Kinesis Video
 Streams"
        ],
        "Effect": "Allow"
    },
    {
        "Action": [
            "kinesisvideo:DescribeStream"
        ],
        "Resource": [
            "Kinesis Video Streams stream ARN"
        ],
        "Effect": "Allow"
    },
    ...
```

외부 데이터 소스에 대한 커넥터에 대한 권한

외부 데이터 소스에 연결하는 함수를 사용하는 구성 요소 유형을 생성하는 경우, 해당 함수를 구현하는 Lambda 함수를 사용할 권한을 서비스 역할에 부여해야 합니다. 구성 요소 유형 및 함수 생성에 대한 자세한 내용은 ???을(를) 참조하십시오.

다음 예제는 서비스 역할에 Lambda 함수를 사용할 수 있는 권한을 부여합니다.

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucket*",
        "s3:GetObject",
        "s3:ListBucket",
        "s3:PutObject"
      ],
      "Resource": [
          "arn:aws:s3:::amzn-s3-demo-bucket",
          "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ]
    },
    {
        "Action": [
            "lambda:invokeFunction"
        ],
        "Resource": [
        "arn:aws:lambda:us-east-1:111122223333:function:example-function"
        ],
        "Effect": "Allow"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:DeleteObject"
      ],
      "Resource": [
        "arn:aws:s3:::*/DO_NOT_DELETE_WORKSPACE_*"
      ]
    }
  ]
}
```

IAM 콘솔, 및 IAM API를 사용하여 역할을 생성하고 정책 AWS CLI및 신뢰 관계를 할당하는 방법에 대한 자세한 내용은 에 권한을 위임할 역할 생성을 참조하세요 AWS 서비스.

Athena 데이터 커넥터를 사용하도록 워크스페이스 IAM 역할을 수정합니다.

AWS IoT TwinMaker Athena 테이블 형식 데이터 커넥터를 사용하려면 AWS IoT TwinMaker 워크스페이스 IAM 역할을 업데이트해야 합니다. 워크스페이스 IAM 역할에 다음 권한을 추가합니다.

> **Note**
>
> 이 IAM 변경은 AWS Glue 및 Amazon S3에 저장된 Athena 테이블 형식 데이터에 대해서만 작동합니다. Athena를 다른 데이터 소스와 함께 사용하려면 Athena에 대한 IAM 역할을 구성해야 합니다. Athena의 ID 및 액세스 관리를 참조하십시오.

```
{
    "Effect": "Allow",
    "Action": [
        "athena:GetQueryExecution",
        "athena:GetQueryResults",
        "athena:GetTableMetadata",
        "athena:GetWorkGroup",
        "athena:StartQueryExecution",
        "athena:StopQueryExecution"
    ],
    "Resource": [
        "athena resouces arn"
    ]
},// Athena permission
{
    "Effect": "Allow",
    "Action": [
        "glue:GetTable",
        "glue:GetTables",
        "glue:GetDatabase",
        "glue:GetDatabases"
    ],
    "Resource": [
        "glue resouces arn"
    ]
},// This is an example for accessing aws glue
{
    "Effect": "Allow",
    "Action": [
        "s3:ListBucket",
```

```
            "s3:GetObject"
        ],
        "Resource": [
            "Amazon S3 data source bucket resources arn"
        ]
    }, // S3 bucket for storing the tabular data.
    {
        "Effect": "Allow",
        "Action": [
            "s3:GetBucketLocation",
            "s3:GetObject",
            "s3:ListBucket",
            "s3:ListBucketMultipartUploads",
            "s3:ListMultipartUploadParts",
            "s3:AbortMultipartUpload",
            "s3:CreateBucket",
            "s3:PutObject",
            "s3:PutBucketPublicAccessBlock"
        ],
        "Resource": [
            "S3 query result bucket resources arn"
        ]
    } // Storing the query results
```

Athena IAM 구성에 대한 자세한 내용은 Athena의 ID 및 액세스 관리를 참조하십시오.

워크스페이스 생성

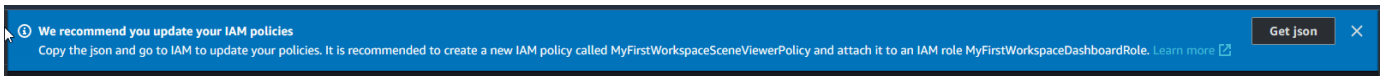
첫 번째 워크스페이스를 생성하고 구성하려면 다음 단계를 사용하십시오.

> **Note**
>
> 이 주제에서는 단일 리소스로 간단한 워크스페이스를 생성하는 방법을 보여줍니다. 리소스가 여러 개인 완전한 기능을 갖춘 워크스페이스의 경우 샘플 Github 리포지토리에서 AWS IoT TwinMaker 샘플 설정을 시도하세요.

1. AWS IoT TwinMaker 콘솔 홈 페이지의 왼쪽 탐색 창에서 워크스페이스를 선택합니다.
2. 워크스페이스 페이지에서 워크스페이스 생성을 선택합니다.
3. 워크스페이스 생성 페이지에서 워크스페이스 이름을 입력합니다.

4. (선택 사항) 워크스페이스에 대한 설명을 추가합니다.
5. S3 리소스에서 S3 버킷 생성을 선택합니다. 이 옵션은가 워크스페이스와 관련된 정보와 리소스를 AWS IoT TwinMaker 저장하는 Amazon S3 버킷을 생성합니다. 각 워크스페이스에는 자체 버킷이 필요합니다.
6. 실행 역할에서 새 역할 자동 생성 또는 이 워크스페이스에 대해 생성한 사용자 지정 IAM 역할을 선택합니다.

   새 역할 자동 생성을 선택하면는 이전 단계에서 지정한 Amazon S3 버킷을 읽고 쓸 수 있는 권한을 포함하여 새 서비스 역할에 다른 AWS 서비스에 액세스할 수 있는 권한을 부여하는 정책을 역할에 AWS IoT TwinMaker 연결합니다. 이 역할에 권한을 할당하는 방법에 대한 자세한 내용은 ???을 참조하십시오.
7. 워크스페이스 생성을 선택합니다. 워크스페이스 페이지 상단에 다음 배너가 나타납니다.



8. Get json을 선택합니다. Grafana 대시보드를 보는 사용자 및 계정에 대해가 AWS IoT TwinMaker 생성한 IAM 역할에 표시되는 IAM 정책을 추가하는 것이 좋습니다. 이 역할의 이름은 *workspace-name* DashboardRole 패턴을 따릅니다. 정책을 생성하여 역할에 연결하는 방법에 대한 지침은 역할 권한 정책 수정(콘솔)을 참조하십시오.

   다음 예에는 대시보드 역할에 추가할 정책이 포함되어 있습니다.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::iottwinmaker-workspace-workspace-name-lower-case-123456789012",
                "arn:aws:s3:::iottwinmaker-workspace-workspace-name-lower-case-123456789012/*"
            ]
        },
        {
```

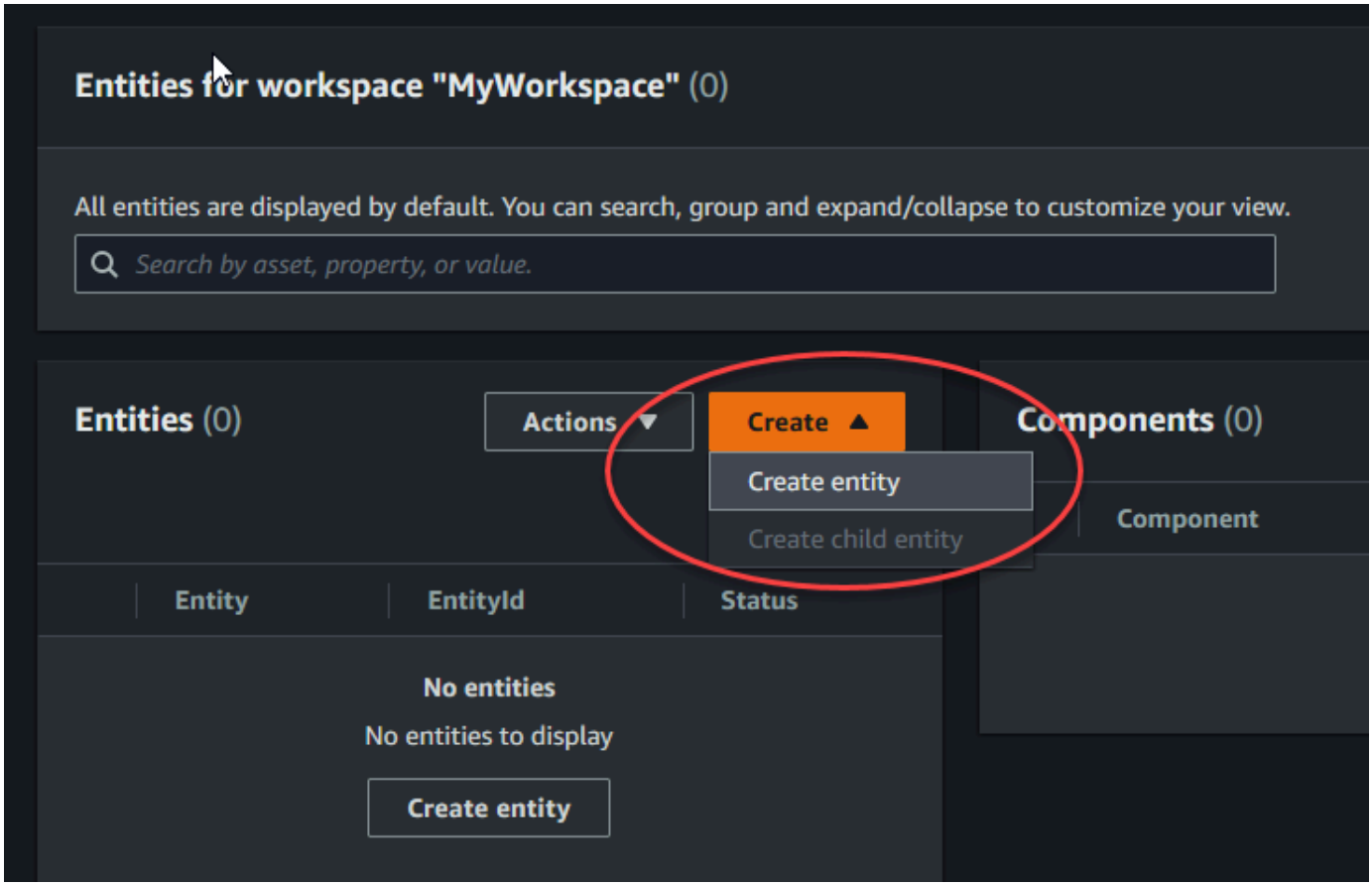
```
            "Effect": "Allow",
            "Action": [
                "iottwinmaker:Get*",
                "iottwinmaker:List*"
            ],
            "Resource": [
                "arn:aws:iottwinmaker:us-
east-1:111122223333:workspace/workspace-name",
                "arn:aws:iottwinmaker:us-
east-1:111122223333:workspace/workspace-name/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "iottwinmaker:ListWorkspaces",
            "Resource": "*"
        }
    ]
}
```

이제 첫 번째 개체를 사용하여 워크스페이스를 위한 데이터 모델을 만들 준비가 되었습니다. 작업 방법에 대한 지침은 첫 번째 개체 생성을(를) 참조하십시오.

첫 번째 개체 생성

첫 번째 개체를 생성하려면 다음 단계를 완료하십시오.

1. 워크스페이스 페이지에서 워크스페이스를 선택한 다음 왼쪽 창에서 개체를 선택합니다.
2. 개체 페이지에서 생성 을 선택한 다음 개체 생성을 선택합니다.



3. 개체 생성 창에서 개체 이름을 입력합니다. 이 예제에서는 **`CookieMixer`** 개체를 사용합니다.
4. (선택 사항) 개체에 대한 설명을 입력합니다.
5. 개체 생성을 선택합니다.

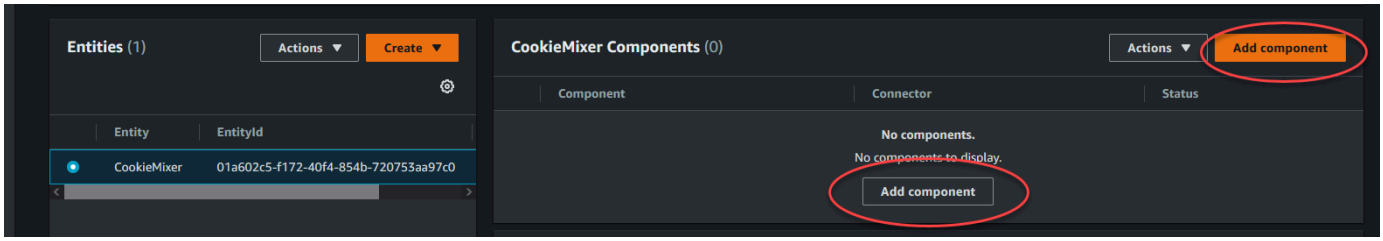
개체에는 워크스페이스의 각 항목에 대한 데이터가 포함됩니다. 구성 요소를 추가하여 엔터티에 데이터를 넣습니다.는 다음과 같은 기본 제공 구성 요소 유형을 AWS IoT TwinMaker 제공합니다.

- 파라미터: 키-값 속성 세트를 추가합니다.
- 문서: 개체에 대한 정보가 포함된 문서의 이름과 URL을 추가합니다.
- 알람: 알람 시계열 데이터 소스에 연결합니다.
- SiteWise 커넥터: AWS IoT SiteWise 자산에 정의된 시계열 속성을 가져옵니다.
- Kinesis Video Streams용 엣지 커넥터 AWS IoT Greengrass: KVS용 엣지 커넥터에서 비디오 데이터를 가져옵니다 AWS IoT Greengrass. 자세한 내용은 AWS IoT TwinMaker 비디오 통합 단원을 참조하십시오.

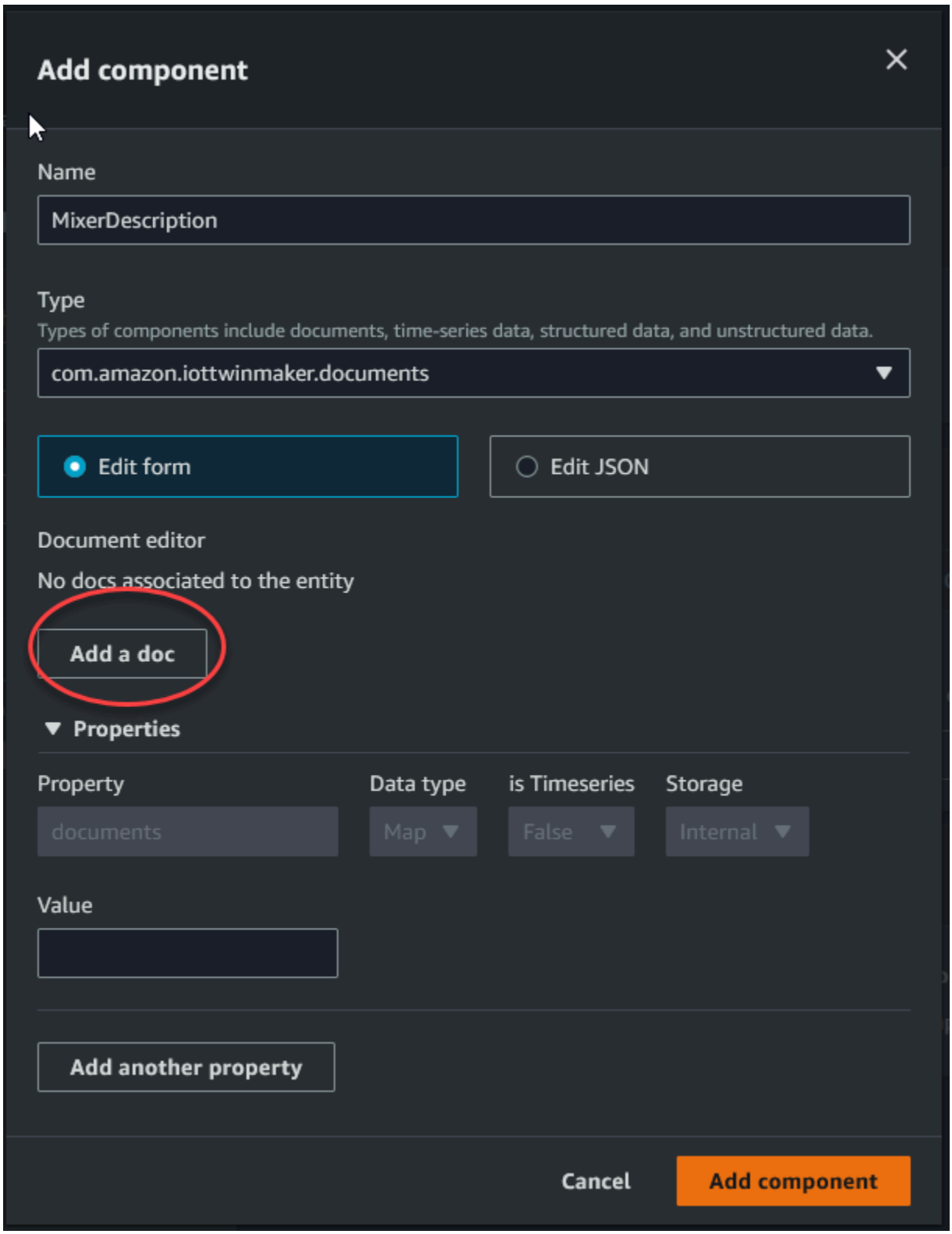
왼쪽 창에서 구성 요소 유형을 선택하여 이러한 구성 요소 유형과 해당 정의를 볼 수 있습니다. 구성 요소 유형 페이지에서 새 구성 요소 유형을 만들 수도 있습니다. 구성 요소 유형 생성에 대한 자세한 내용은 구성 요소 유형 사용 및 생성을(를) 참조하십시오.

이 예제에서는 개체에 대한 설명 정보를 추가하는 간단한 문서 구성 요소를 만듭니다.

1. 개체 페이지에서 개체를 선택한 다음 구성 요소 추가를 선택합니다.



2. 구성 요소 추가 창에서 구성 요소 이름을 입력합니다. 이 예제에서는 쿠키 믹서 개체를 사용하므로 이름 필드에 **MixerDescription**을(를) 입력합니다.



3. 문서 추가를 선택한 다음 문서 이름 및 외부 URL의 값을 입력합니다. 문서 구성 요소를 사용하면 개체에 대한 중요한 정보가 포함된 외부 URLs 목록을 저장할 수 있습니다.
4. 구성 요소 추가를 선택합니다.

이제 첫 장면을 만들 준비가 되었습니다. 작업 방법에 대한 지침은 AWS IoT TwinMaker 장면 생성 및 편집을(를) 참조하십시오.

AWS 계정 설정

이 없는 경우 다음 단계를 AWS 계정완료하여 생성합니다.

에 가입하려면 AWS 계정

1. https://portal.aws.amazon.com/billing/signup을 엽니다.
2. 온라인 지시 사항을 따르세요.

   등록 절차 중 전화 또는 텍스트 메시지를 받고 전화 키패드로 확인 코드를 입력하는 과정이 있습니다.

   에 가입하면 AWS 계정AWS 계정 루트 사용자이 생성됩니다. 루트 사용자에게는 계정의 모든 AWS 서비스 및 리소스에 액세스할 권한이 있습니다. 보안 모범 사례는 사용자에게 관리 액세스 권한을 할당하고, 루트 사용자만 사용하여 루트 사용자 액세스 권한이 필요한 작업을 수행하는 것입니다.

구성 요소 유형 사용 및 생성

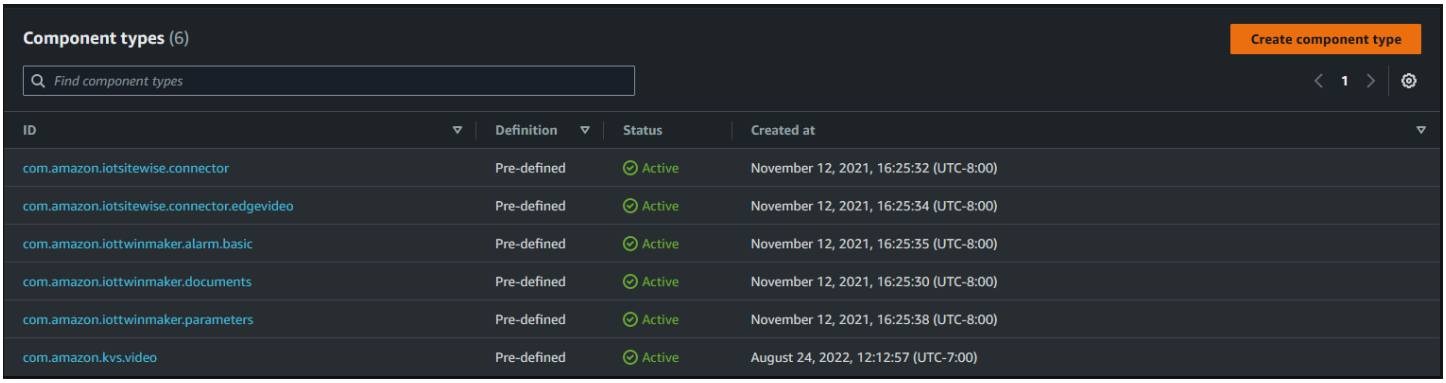
이 항목에서는 AWS IoT TwinMaker 구성 요소 유형을 만드는 데 사용하는 값과 구조를 안내합니다. CreateComponentTypeAPI에 전달하거나 AWS IoT TwinMaker 콘솔의 구성 요소 유형 편집기를 사용하여 전달할 수 있는 요청 객체를 생성하는 방법을 보여 줍니다.

구성 요소는 관련 개체의 속성 및 데이터에 대한 컨텍스트를 제공합니다.

기본 설정 구성 요소 유형

AWS IoT TwinMaker 콘솔에서 작업 영역을 선택한 다음 왼쪽 창에서 구성 요소 유형을 선택하면 다음과 같은 구성 요소 유형이 표시됩니다.

- com.amazon.iotsitewise.resourcesync: 자산과 자산 모델을 자동으로 동기화하고 이를 엔티티 구성 요소 및 구성 요소 유형으로 변환하는 구성 요소 유형입니다. AWS IoT SiteWise AWS IoT TwinMaker 자산 동기화 사용에 대한 자세한 내용은 AWS IoT SiteWise 자산 AWS IoT SiteWise 동기화를 참조하십시오.
- com.amazon.iottwinmaker.alarm.basic: 외부 소스에서 개체로 경보 데이터를 가져오는 기본 경보 구성 요소입니다. 이 구성 요소에는 특정 데이터 소스에 연결하는 함수가 포함되어 있지 않습니다. 즉, 경보 구성 요소는 추상적이며 데이터 소스를 지정하는 다른 구성 요소 유형과 해당 소스에서 읽는 함수에 상속될 수 있습니다.
- com.amazon.iottwinmaker.documents: 개체에 대한 정보가 포함된 문서의 제목을 URL에 간단히 매핑하는 것입니다.
- com.amazon.iotsitewise.connector.edgevideo: Kinesis Video Streams용 에지 커넥터 구성 요소를 사용하여 IoT 디바이스에서 비디오를 엔티티로 가져오는 구성 요소입니다. AWS IoT Greengrass Kinesis Video AWS IoT Greengrass Streams용 에지 AWS IoT TwinMaker 커넥터 구성 요소는 구성 요소가 아니라 IoT 장치에 로컬로 배포되는 AWS IoT Greengrass 사전 빌드된 구성 요소입니다.
- com.amazon.iotsitewise.connector: AWS IoT SiteWise 데이터를 개체로 가져오는 구성 요소입니다.
- com.amazon.iottwinmaker.parameters: 개체에 정적 키-값 쌍을 추가하는 구성 요소입니다.
- com.amazon.kvs.video: Kinesis Video Streams에서 비디오를 엔티티로 가져오는 구성 요소입니다. AWS IoT TwinMaker



AWS IoT TwinMaker 구성 요소 유형의 핵심 기능

다음 목록에서는 구성 요소 유형의 핵심 기능에 대해 설명합니다.

- 속성 정의: PropertyDefinitionRequest객체는 씬 컴포저에서 채울 수 있는 속성을 정의하거나 외부 데이터 소스에서 가져온 데이터로 채울 수 있는 속성을 정의합니다. 설정한 정적 속성은 저장되어 있습니다. AWS IoT TwinMaker시계열 속성 및 데이터 소스에서 가져온 기타 속성은 외부에 저장됩니다.

  문자열 내에서 `PropertyDefinitionRequest` 맵에 속성 정의를 지정합니다. 각 문자열은 맵에 대해 고유해야 합니다.

- 함수: FunctionRequest객체는 외부 데이터 소스에서 읽고 잠재적으로 외부 데이터 소스에 쓰는 Lambda 함수를 지정합니다.

  값이 외부에 저장되어 있는 속성을 포함하지만 값을 검색하는 해당 함수가 없는 구성 요소 유형은 추상 구성 요소 유형입니다. 추상 구성 요소 유형에서 구체적인 구성 요소 유형을 확장할 수 있습니다. 개체에는 추상 구성 요소 유형을 추가할 수 없습니다. 장면 컴포저에는 나타나지 않습니다.

  문자열 내에 `FunctionRequest` 매핑할 함수를 지정합니다. 문자열은 다음과 같은 사전 정의된 함수 유형 중 하나만 지정해야 합니다.

  - `dataReader`: 외부 소스에서 데이터를 가져오는 함수입니다.
  - `dataReaderByEntity`: 외부 소스에서 데이터를 가져오는 함수입니다.

    이 유형의 데이터 리더를 사용하는 경우 GetPropertyValueHistoryAPI 작업은 이 구성 요소 유형의 속성에 대한 엔티티별 쿼리만 지원합니다. (`componentName` + `entityId`에 대한 속성 값 기록만 요청할 수 있습니다.)
  - `dataReaderByComponentType`: 외부 소스에서 데이터를 가져오는 함수입니다.

이 유형의 데이터 리더를 사용하는 경우 GetPropertyValueHistoryAPI 작업은 이 구성 요소 유형의 속성에 대한 개체 간 쿼리만 지원합니다. (componentTypeId에 대한 속성 값 기록만 요청할 수 있습니다.)

- `dataWriter`: 외부 소스에 데이터를 쓰는 함수입니다.
- `schemaInitializer`: 구성 요소 유형이 포함된 개체를 만들 때마다 속성값을 자동으로 초기화하는 함수입니다.

비추상 구성 요소 유형에는 세 가지 유형의 데이터 판독기 함수 중 하나가 필요합니다.

경보를 포함하여 타임스트림 텔레메트리 구성 요소를 구현하는 Lambda 함수의 예는 AWS IoT TwinMaker 샘플의 데이터 리더를 참조하십시오.

> **Note**
>
> 경보 커넥터는 추상 경보 구성 요소 유형을 상속하므로 Lambda 함수는 `alarm_key`값을 반환해야 합니다. 이 값을 반환하지 않으면 Grafana는 이 값을 경보로 인식하지 않습니다. 이는 경보를 반환하는 모든 구성 요소에 필요합니다.

- 상속: 구성 요소 유형은 상속을 통해 코드 재사용성을 높입니다. 구성 요소 유형은 최대 10개의 상위 구성 요소 유형을 상속할 수 있습니다.

  `extendsFrom`매개 변수를 사용하여 구성 요소 유형이 속성과 함수를 상속하는 구성 요소 유형을 지정할 수 있습니다.
- isSingleton: 일부 구성 요소에는 개체에 두 번 이상 포함할 수 없는 위치 좌표와 같은 속성이 포함되어 있습니다. 구성 요소 유형을 개체에 한 번만 포함할 수 있다는 것을 알리도록 `isSingleton` 파라미터 값을 `true`로 설정합니다.

속성 정의 생성

다음 표는 `PropertyDefinitionRequest`의 파라미터를 설명합니다.

파라미터	설명
`isExternalId`	속성이 외부에 저장된 속성 값의 고유 식별자(예: AWS IoT SiteWise 자산 ID) 인지 여부를 지정하는 불리언입니다.

파라미터	설명
	이 속성의 기본값은 `false`입니다.
`isStoredExternally`	속성이 외부에 저장되는지 여부를 지정하는 부울 값입니다. 이 속성의 기본값은 `false`입니다.
`isTimeSeries`	속성이 시계열 데이터로 구성되는지 여부를 지정하는 부울 값입니다. 이 속성의 기본값은 `false`입니다.
`isRequiredInEntity`	구성 요소 유형을 사용하는 개체에서 속성에 값이 있어야 하는지 여부를 지정하는 불리언입니다.
`dataType`	속성의 데이터 유형 (예: 문자열, 맵, 목록, 측정 단위) 을 지정하는 DataType객체입니다.
`defaultValue`	속성의 기본값을 지정하는 DataValue객체입니다.
`configuration`	외부 데이터 소스에 연결하는 데 필요한 추가 정보를 지정하는 string-to-string 맵입니다.

함수 생성

다음 표는 `FunctionRequest`의 파라미터를 설명합니다.

파라미터	설명
`implementedBy`	외부 데이터 소스에 연결하는 Lambda 함수를 지정하는 DataConnector객체입니다.
`requiredProperties`	함수가 외부 데이터 소스에서 읽고 외부 데이터 소스에 쓰기 위해 필요한 속성 목록입니다.

파라미터	설명
scope	함수의 범위입니다. 범위가 전체 작업 영역에 적용되는 함수에 대해 Workspace 을(를) 사용합니다. 구성 요소가 포함된 개체로 범위가 제한된 함수에 대해 Entity을(를) 사용합니다.

구성 요소 유형을 만들고 확장하는 방법을 보여주는 예제는 ???을(를) 참조하십시오.

구성 요소 유형 예

이 주제에는 구성 요소 유형의 주요 개념을 구현하는 방법을 보여주는 예제가 포함되어 있습니다.

경보(요약)

다음 예는 콘솔에 나타나는 추상 경보 구성 요소 유형입니다. AWS IoT TwinMaker 여기에는 functions 목록이 포함되며 이 목록은 dataReader(으)로 구성되며 여기에는 implementedBy값이 없습니다.

```
{
  "componentTypeId": "com.example.alarm.basic:1",
  "workspaceId": "MyWorkspace",
  "description": "Abstract alarm component type",
  "functions": {
    "dataReader": {
         "isInherited": false
    }
  },
  "isSingleton": false,
  "propertyDefinitions": {
    "alarm_key": {
      "dataType": { "type": "STRING" },
      "isExternalId": true,
      "isRequiredInEntity": true,
      "isStoredExternally": false,
      "isTimeSeries": false
    },
    "alarm_status": {
      "dataType": {
```

```
        "allowedValues": [
          {
            "stringValue": "ACTIVE"
          },
          {
            "stringValue": "SNOOZE_DISABLED"
          },
          {
            "stringValue": "ACKNOWLEDGED"
          },
          {
            "stringValue": "NORMAL"
          }
        ],
        "type": "STRING"
      },
      "isRequiredInEntity": false,
      "isStoredExternally": true,
      "isTimeSeries": true
    }
  }
}
```

참고:

componentTypeId 및 workspaceID에 대한 값은 필수입니다. componentTypeId의 값은 작업 영역별로 고유해야 합니다. alarm_key의 값은 함수가 외부 소스에서 경보 데이터를 검색하는 데 사용할 수 있는 고유 식별자입니다. 키 값은 필수이며 저장되어 AWS IoT TwinMaker있습니다. alarm_status 시계열 값은 외부 소스에 저장됩니다.

AWS IoT TwinMaker 샘플에서 더 많은 예를 사용할 수 있습니다.

Timestream 원격측정

다음 예는 외부 소스에서 특정 유형의 구성 요소 (예: 경보 또는 쿠키 믹서) 에 대한 원격 분석 데이터를 검색하는 간단한 구성 요소 유형입니다. 구성 요소 유형이 상속하는 Lambda 함수를 지정합니다.

```
{
    "componentTypeId": "com.example.timestream-telemetry",
    "workspaceId": "MyWorkspace",
```

```
    "functions": {
        "dataReader": {
            "implementedBy": {
                "lambda": {
                    "arn": "lambdaArn"
                }
            }
        }
    },
    "propertyDefinitions": {
        "telemetryType": {
            "dataType": { "type": "STRING" },
            "isExternalId": false,
            "isStoredExternally": false,
            "isTimeSeries": false,
            "isRequiredInEntity": true
        },
        "telemetryId": {
            "dataType": { "type": "STRING" },
            "isExternalId": false,
            "isStoredExternally": false,
            "isTimeSeries": false,
            "isRequiredInEntity": true
        }
    }
}
```

경보(추상 경보에서 상속)

다음 예제는 추상 경보와 타임스트림 텔레메트리 구성 요소 유형을 모두 상속합니다. 경보 데이터를 검색하는 자체 Lambda 함수를 지정합니다.

```
{
    "componentTypeId": "com.example.cookiefactory.alarm",
    "workspaceId": "MyWorkspace",
    "extendsFrom": [
        "com.example.timestream-telemetry",
        "com.amazon.iottwinmaker.alarm.basic"
    ],
    "propertyDefinitions": {
        "telemetryType": {
```

```
            "defaultValue": {
                "stringValue": "Alarm"
            }
        }
    },
    "functions": {
        "dataReader": {
            "implementedBy": {
                "lambda": {
                    "arn": "lambdaArn"
                }
            }
        }
    }
}
```

> **Note**
>
> 경보 커넥터는 추상 경보 구성 요소 유형을 상속하므로 Lambda 함수는 `alarm_key`값을 반환해야 합니다. 이 값을 반환하지 않으면 Grafana는 이 값을 경보로 인식하지 않습니다. 이는 경보를 반환하는 모든 구성 요소에 필요합니다.

장비 예

이 단원의 예제에서는 잠재적 장비를 모델링하는 방법을 보여줍니다. 이 예제를 사용하여 자체 프로세스에서 장비를 모델링하는 방법에 대한 아이디어를 얻을 수 있습니다.

쿠키 믹서

다음 예제는 타임스트림 텔레메트리 구성 요소 유형으로 부터 상속합니다. 쿠키 믹서의 회전 속도 및 온도에 대한 추가 시계열 속성을 지정합니다.

```
{
    "componentTypeId": "com.example.cookiefactory.mixer",
    "workspaceId": "MyWorkspace",
    "extendsFrom": [
        "com.example.timestream-telemetry"
    ],
```

```
    "propertyDefinitions": {
        "telemetryType": {
            "defaultValue" : { "stringValue": "Mixer" }
        },
        "RPM": {
            "dataType": { "type": "DOUBLE" },
            "isTimeSeries": true,
            "isStoredExternally": true
        },
        "Temperature": {
            "dataType": { "type": "DOUBLE" },
            "isTimeSeries": true,
            "isStoredExternally": true
        }
    }
}
```

물 탱크

다음 예제는 타임스트림 텔레메트리 구성 요소 유형으로 부터 상속합니다. 물 탱크의 부피 및 유량에 대한 추가 시계열 속성을 지정합니다.

```
{
    "componentTypeId": "com.example.cookiefactory.watertank",
    "workspaceId": "MyWorkspace",
    "extendsFrom": [
        "com.example.timestream-telemetry"
    ],
    "propertyDefinitions": {
        "telemetryType": {
            "defaultValue" : { "stringValue": "WaterTank" }
        },
        "tankVolume1": {
            "dataType": { "type": "DOUBLE" },
            "isTimeSeries": true,
            "isStoredExternally": true
        },
        "tankVolume2": {
            "dataType": { "type": "DOUBLE" },
            "isTimeSeries": true,
            "isStoredExternally": true
```

```
        },
        "flowRate1": {
            "dataType": { "type": "DOUBLE" },
            "isTimeSeries": true,
            "isStoredExternally": true
        },
        "flowrate2": {
            "dataType": { "type": "DOUBLE" },
            "isTimeSeries": true,
            "isStoredExternally": true
        }
    }
}
```

스페이스 위치

다음 예제에는 값이 저장되어 있는 속성이 포함되어 있습니다. AWS IoT TwinMaker값은 사용자가 지정하고 내부적으로 저장하기 때문에 해당 값을 검색하는 데 함수가 필요하지 않습니다. 또한 이 예제에서는 RELATIONSHIP 데이터 유형을 사용하여 다른 구성 요소 유형과의 관계를 지정합니다.

이 구성 요소는 디지털 트윈에 컨텍스트를 추가할 수 있는 간단한 메커니즘을 제공합니다. 이를 사용하여 사물의 위치를 나타내는 메타데이터를 추가할 수 있습니다. 또한 이 정보를 어떤 카메라가 장비나 스페이스를 볼 수 있는지 결정하는 데 사용되는 로직이나 특정 위치로 사람을 파견하는 방법을 파악하는 데 사용할 수 있습니다.

```
{
    "componentTypeId": "com.example.cookiefactory.space",
    "workspaceId": "MyWorkspace",
    "propertyDefinitions": {
        "position":  {"dataType": {"nestedType": {"type": "DOUBLE"},"type": "LIST"}},
        "rotation":  {"dataType": {"nestedType": {"type": "DOUBLE"},"type": "LIST"}},
        "bounds":  {"dataType": {"nestedType": {"type": "DOUBLE"},"type": "LIST"}},
        "parent_space" : { "dataType": {"type": "RELATIONSHIP"}}
    }
}
```

AWS IoT TwinMaker 대량 작업

metadataTransferJob을 사용하여 AWS IoT TwinMaker 리소스를 대규모로 전송하고 관리합니다. metadataTransferJob을 사용하면 대량 작업을 수행하고 AWS IoT TwinMaker 및 Amazon S3 간에 리소스를 전송할 수 AWS IoT SiteWise 있습니다.

다음 시나리오에서 대량 작업을 사용할 수 있습니다.

- 개발 계정에서 프로덕션 계정으로 마이그레이션하는 등 계정 간 자산 및 데이터의 대량 마이그레이션.
- 대규모 자산 업로드 및 편집과 같은 대규모 AWS IoT 자산 관리.
- 자산을 AWS IoT TwinMaker 및 로 대량 가져옵니다 AWS IoT SiteWise.
- `revit` 또는 파일과 같은 기존 온톨로지 `BIM` 파일에서 AWS IoT TwinMaker 엔터티를 대량으로 가져옵니다.

주제

주요 개념 및 용어

AWS IoT TwinMaker 대량 작업은 다음 개념과 용어를 사용합니다.

- 가져오기: 리소스를 워크스페이스로 이동하는 작업입니다 AWS IoT TwinMaker . 예를 들어 로컬 파일, Amazon S3 버킷의 파일 또는에서 AWS IoT TwinMaker 워크스페이스 AWS IoT SiteWise 로.
- 내보내기: 워크스페이스에서 AWS IoT TwinMaker 로컬 시스템 또는 Amazon S3 버킷으로 리소스를 이동하는 작업입니다.
- 소스: 리소스를 이동할 시작 위치입니다.

  예를 들어 Amazon S3 버킷은 가져오기 소스이고 AWS IoT TwinMaker 워크스페이스는 내보내기 소스입니다.
- 대상: 리소스를 이동할 위치입니다.

예를 들어 Amazon S3 버킷은 내보내기 대상이고 AWS IoT TwinMaker 워크스페이스는 가져오기 대상입니다.

- AWS IoT SiteWise 스키마: 리소스를 가져오고 내보내는 데 사용되는 스키마입니다 AWS IoT SiteWise.
- AWS IoT TwinMaker 스키마: 리소스를 가져오고 내보내는 데 사용되는 스키마입니다 AWS IoT TwinMaker.
- AWS IoT TwinMaker 최상위 리소스: 기존 APIs. 특히 개체 또는 ComponentType입니다.
- AWS IoT TwinMaker 하위 수준 리소스: 메타데이터 정의에 사용되는 중첩 리소스 유형입니다. 특히 구성 요소입니다.
- 메타데이터: AWS IoT SiteWise 및 AWS IoT TwinMaker 리소스를 성공적으로 가져오거나 내보내는 데 필요한 주요 정보입니다.
- metadataTransferJob: `CreateMetadataTransferJob`을 실행할 때 생성된 객체입니다.

AWS IoT TwinMaker metadataTransferJob 기능

이 주제에서는 일괄 작업을 실행할 때 AWS IoT TwinMaker 의 동작, 즉 metadataTransferJob이 처리되는 방식에 대해 설명합니다. 또한 리소스를 전송하는 데 필요한 메타데이터를 사용하여 스키마를 정의하는 방법을 설명합니다. AWS IoT TwinMaker bulk 작업은 다음 기능을 지원합니다.

- 최상위 리소스 생성 또는 대체: AWS IoT TwinMaker 는 새 리소스를 생성하거나 리소스 ID로 고유하게 식별되는 모든 기존 리소스를 대체합니다.

  예를 들어 엔터티가 시스템에 있는 경우 엔터티 정의는 `Entity` 키 아래의 템플릿에 정의된 새 엔터티 정의로 대체됩니다.
- 하위 리소스 생성 또는 교체:

  EntityComponent 수준에서는 구성 요소만 생성하거나 교체할 수 있습니다. 개체가 이미 있어야 합니다. 그렇지 않으면 작업이 ValidationException을 생성합니다.

  속성 또는 관계 수준에서는 속성 또는 관계만 생성하거나 바꿀 수 있으며 포함된 EntityComponent가 이미 있어야 합니다.
- 하위 리소스 삭제:

  AWS IoT TwinMaker 는 하위 리소스 삭제도 지원합니다. 하위 리소스는 구성 요소, 속성 또는 관계일 수 있습니다.

구성 요소를 삭제하려면 개체 수준에서 삭제해야 합니다.

속성 또는 관계를 삭제하려면 Entity 또는 EntityComponent 수준에서 삭제해야 합니다.

하위 리소스를 삭제하려면 상위 수준 리소스를 업데이트하고 하위 리소스의 정의를 생략합니다.

- 최상위 리소스 삭제 없음: AWS IoT TwinMaker 는 최상위 리소스를 삭제하지 않습니다. 최상위 리소스는 개체 또는 ComponentType을 나타냅니다.
- 하나의 템플릿에 동일한 최상위 리소스에 대한 하위 리소스 정의가 없습니다.

  동일한 템플릿에서 동일한 개체의 전체 개체 정의 및 하위 리소스(예: 속성) 정의를 제공할 수 없습니다.

  entityId가 Entity에서 사용되는 경우 Entity, EntityComponent, 속성 또는 관계에 동일한 ID를 사용할 수 없습니다.

  EntityComponent entityId 또는 componentName 조합을 사용하는 경우 EntityComponent, 속성 또는 관계에서 동일한 조합을 사용할 수 없습니다.

  entityId, componentName, propertyName 조합을 속성 또는 관계에 사용하는 경우 속성 또는 관계에 동일한 조합을 사용할 수 없습니다.
- ExternalId는 선택 사항입니다 AWS IoT TwinMaker. ExternalId를 사용하여 리소스를 식별할 수 있습니다.

대량 가져오기 및 내보내기 작업 수행

이 주제에서는 대량 가져오기 및 내보내기 작업을 수행하는 방법과 전송 작업의 오류를 처리하는 방법을 다룹니다. CLI 명령을 사용한 전송 작업의 예를 제공합니다.

AWS IoT TwinMaker API 참조에는 CreateMetadataTransferJob 및 기타 API 작업에 대한 정보가 포함되어 있습니다.

주제

- metadataTransferJob 사전 조건
- IAM 권한
- 대량 작업 실행
- 오류 처리
- 메타데이터 템플릿 가져오기

- AWS IoT TwinMaker metadataTransferJob 예제

metadataTransferJob 사전 조건

metadataTransferJob을 실행하기 전에 다음 사전 조건을 완료하세요.

- AWS IoT TwinMaker 워크스페이스를 생성합니다. 워크스페이스는 metadataTransferJob의 가져오기 대상 또는 내보내기 소스일 수 있습니다. 워크스페이스 생성에 대한 자세한 내용은 섹션을 참조하세요워크스페이스 생성.
- 리소스를 저장할 Amazon S3 버킷을 생성합니다. Amazon S3 사용에 대한 자세한 내용은 Amazon S3란 무엇입니까?를 참조하세요.

IAM 권한

대량 작업을 수행할 때는 Amazon S3 AWS IoT SiteWise, AWS IoT TwinMaker및 로컬 시스템 간에 AWS 리소스를 교환할 수 있는 권한이 있는 IAM 정책을 생성해야 합니다. IAM 정책 생성에 대한 자세한 내용은 IAM 정책 생성을 참조하세요.

AWS IoT TwinMaker AWS IoT SiteWise 및 Amazon S3에 대한 정책 설명은 다음과 같습니다.

- AWS IoT TwinMaker 정책:

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetBucketLocation",
            "s3:ListBucket",
            "s3:AbortMultipartUpload",
            "s3:ListBucketMultipartUploads",
            "s3:ListMultipartUploadParts"
        ],
        "Resource": "*"
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "iottwinmaker:GetWorkspace",
                "iottwinmaker:CreateEntity",
                "iottwinmaker:GetEntity",
                "iottwinmaker:UpdateEntity",
                "iottwinmaker:GetComponentType",
                "iottwinmaker:CreateComponentType",
                "iottwinmaker:UpdateComponentType",
                "iottwinmaker:ListEntities",
                "iottwinmaker:ListComponentTypes",
                "iottwinmaker:ListTagsForResource",
                "iottwinmaker:TagResource",
                "iottwinmaker:UntagResource"
            ],
            "Resource": "*"
        }
    ]
}
```

- AWS IoT SiteWise 정책:

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetBucketLocation",
            "s3:ListBucket",
            "s3:AbortMultipartUpload",
            "s3:ListBucketMultipartUploads",
            "s3:ListMultipartUploadParts"
        ],
        "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:CreateAsset",
                "iotsitewise:CreateAssetModel",
```

```
                "iotsitewise:UpdateAsset",
                "iotsitewise:UpdateAssetModel",
                "iotsitewise:UpdateAssetProperty",
                "iotsitewise:ListAssets",
                "iotsitewise:ListAssetModels",
                "iotsitewise:ListAssetProperties",
                "iotsitewise:ListAssetModelProperties",
                "iotsitewise:ListAssociatedAssets",
                "iotsitewise:DescribeAsset",
                "iotsitewise:DescribeAssetModel",
                "iotsitewise:DescribeAssetProperty",
                "iotsitewise:AssociateAssets",
                "iotsitewise:DisassociateAssets",
                "iotsitewise:AssociateTimeSeriesToAssetProperty",
                "iotsitewise:DisassociateTimeSeriesFromAssetProperty",
                "iotsitewise:BatchPutAssetPropertyValue",
                "iotsitewise:BatchGetAssetPropertyValue",
                "iotsitewise:TagResource",
                "iotsitewise:UntagResource",
                "iotsitewise:ListTagsForResource"
            ],
            "Resource": "*"
        }
    ]
}
```

- Amazon S3 정책:

```
{
    "Effect": "Allow",
    "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:GetBucketLocation",
        "s3:ListBucket",
        "s3:AbortMultipartUpload",
        "s3:ListBucketMultipartUploads",
        "s3:ListMultipartUploadParts"
    ],
    "Resource": "*"
}
```

또는 단일 Amazon S3 버킷에만 액세스하도록 Amazon S3 정책의 범위를 지정할 수 있습니다. 다음 정책을 참조하세요.

Amazon S3 단일 버킷 범위 지정 정책

```
{
    "Effect": "Allow",
    "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:GetBucketLocation",
        "s3:ListBucket",
        "s3:AbortMultipartUpload",
        "s3:ListBucketMultipartUploads",
        "s3:ListMultipartUploadParts"
    ],
    "Resource": [
        "arn:aws:s3:::bucket name",
        "arn:aws:s3:::bucket name/*"
    ]
}
```

metadataTransferJob에 대한 액세스 제어 설정

사용자가 액세스할 수 있는 작업의 종류를 제어하려면 호출에 사용되는 역할에 다음 IAM 정책을 추가합니다 AWS IoT TwinMaker.

> **Note**
>
> 이 정책은 Amazon S3와 리소스를 주고 받는 AWS IoT TwinMaker 가져오기 및 내보내기 작업에 대한 액세스만 허용합니다.

```
{
    "Effect": "Allow",
    "Action": [
        "iottwinmaker:*DataTransferJob*"
    ],
    "Resource": "*",
    "Condition": {
```

```
        "StringLikeIfExists": {
            "iottwinmaker:sourceType": [
                "s3",
                "iottwinmaker"
            ],
            "iottwinmaker:destinationType": [
                "iottwinmaker",
                "s3"
            ]
        }
    }
}
```

대량 작업 실행

이 섹션에서는 대량 가져오기 및 내보내기 작업을 수행하는 방법을 다룹니다.

Amazon S3에서 로 데이터 가져오기 AWS IoT TwinMaker

1. AWS IoT TwinMaker metadataTransferJob 스키마를 사용하여 전송할 리소스를 지정합니다. 스키마 파일을 생성하여 Amazon S3 버킷에 저장합니다.

   스키마 예제는 단원을 참조하십시오메타데이터 템플릿 가져오기.
2. 요청 본문을 생성하고 JSON 파일로 저장합니다. 요청 본문은 전송 작업의 소스와 대상을 지정합니다. Amazon S3 버킷을 소스로 지정하고 AWS IoT TwinMaker 워크스페이스를 대상으로 지정해야 합니다.

   다음은 요청 본문의 예입니다.

   ```
   {
       "metadataTransferJobId": "your-transfer-job-Id",
       "sources": [{
           "type": "s3",
           "s3Configuration": {
               "location": "arn:aws:s3:::amzn-s3-demo-bucket/your_import_data.json"
           }
       }],
       "destination": {
           "type": "iottwinmaker",
           "iotTwinMakerConfiguration": {
               "workspace": "arn:aws:iottwinmaker:us-
   east-1:111122223333:workspace/your-worksapce-name"
   ```

```
        }
    }
}
```

요청 본문에 지정한 파일 이름을 기록합니다. 다음 단계에서 필요합니다. 이 예제에서 요청 본문의 이름은 입니다`createMetadataTransferJobImport.json`.

3. 다음 CLI 명령을 실행하여를 호출합니다`CreateMetadataTransferJob`(input-json 파일 이름을 요청 본문에 지정한 이름으로 바꿉니다).

```
aws iottwinmaker create-metadata-transfer-job --region us-east-1 \
--cli-input-json file://createMetadataTransferJobImport.json
```

그러면 metadataTransferJob이 생성되고 선택한 리소스 전송 프로세스가 시작됩니다.

에서 Amazon S3 AWS IoT TwinMaker 로 데이터 내보내기

1. 적절한 필터를 사용하여 JSON 요청 본문을 생성하여 내보낼 리소스를 선택합니다. 이 예제에서는 다음을 사용합니다.

```
{
    "metadataTransferJobId": "your-transfer-job-Id",
    "sources": [{
        "type": "iottwinmaker",
        "iotTwinMakerConfiguration": {
            "workspace": "arn:aws:iottwinmaker:us-
east-1:111122223333:workspace/your-workspace-name",
            "filters": [{
                "filterByEntity": {
                    "entityId": "parent"
                }},
                {
                "filterByEntity": {
                    "entityId": "child"
                }},
                {
                "filterByComponentType": {
                    "componentTypeId": "component.type.minimal"
                }}
            ]
        }
```

```
    }],
    "destination": {
        "type": "s3",
        "s3Configuration": {
            "location": "arn:aws:s3:::amzn-s3-demo-bucket"
        }
    }
}
```

`filters` 배열을 사용하면 내보낼 리소스를 지정할 수 있습니다. 이 예제에서는 `entity`, 및를 기준으로 필터링합니다`componentType`.

워크 AWS IoT TwinMaker 스페이스를 소스로 지정하고 Amazon S3 버킷을 메타데이터 전송 작업의 대상으로 지정해야 합니다.

요청 본문을 저장하고 파일 이름을 기록합니다. 다음 단계에서 필요합니다. 이 예제에서는 요청 본문의 이름을 지정했습니다`createMetadataTransferJobExport.json`.

2. 다음 CLI 명령을 실행하여를 호출합니다`CreateMetadataTransferJob`(input-json 파일 이름을 요청 본문에 지정한 이름으로 바꿉니다).

```
aws iottwinmaker create-metadata-transfer-job --region us-east-1 \
--cli-input-json file://createMetadataTransferJobExport.json
```

그러면 metadataTransferJob이 생성되고 선택한 리소스 전송 프로세스가 시작됩니다.

전송 작업의 상태를 확인하거나 업데이트하려면 다음 명령을 사용합니다.

- 작업을 취소하려면 CancelMetadataTransferJob API 작업을 사용합니다. CancelMetadataTransferJob을 호출하면 API는 실행 중인 metadataTransferJob만 취소하며 이미 내보내거나 가져온 리소스는이 API 호출의 영향을 받지 않습니다.
- 특정 작업에 대한 정보를 검색하려면 GetMetadataTransferJob API 작업을 사용합니다.

  또는 다음 CLI 명령을 사용하여 기존 전송 작업에서 GetMetadataTransferJob을 호출할 수 있습니다.

```
aws iottwinmaker get-metadata-transfer-job --job-id ExistingJobId
```

  존재하지 않는 AWS IoT TwinMaker 가져오기 또는 내보내기 작업에서 GetMetadataTransferJob을 호출하면 응답에 `ResourceNotFoundException` 오류가 발생합니다.

- 현재 작업을 나열하려면 ListMetadataTransferJobs API 작업을 사용합니다.

  다음은를 destinationType AWS IoT TwinMaker 으로,를 sourceType으로 사용하여 ListMetadataTransferJobss3를 호출하는 CLI 예제입니다. sourceType

```
aws iottwinmaker list-metadata-transfer-jobs --destination-type iottwinmaker --
source-type s3
```

> **Note**
>
> 가져오기 또는 내보내기 작업의 소스 및 대상과 일치하도록 sourceType 및 destinationType 파라미터의 값을 변경할 수 있습니다.

이러한 API 작업을 호출하는 CLI 명령의 자세한 예는 섹션을 참조하세요AWS IoT TwinMaker metadataTransferJob 예제.

전송 작업 중에 오류가 발생하면 섹션을 참조하세요오류 처리.

오류 처리

전송 작업을 생성하고 실행한 후 GetMetadataTransferJob을 호출하여 발생한 오류를 진단할 수 있습니다.

```
aws iottwinmaker get-metadata-transfer-job \
--metadata-transfer-job-id your_metadata_transfer_job_id \
--region us-east-1
```

작업 상태가 로 바뀌는 COMPLETED것을 확인하면 작업 결과를 확인할 수 있습니다. GetMetadataTransferJob은 다음 필드가 MetadataTransferJobProgress 포함된 라는 객체를 반환합니다.

- failedCount: 전송 프로세스 중에 실패한 리소스 수를 나타냅니다.
- skippedCount: 전송 프로세스 중에 건너뛴 리소스 수를 나타냅니다.
- succeededCount: 전송 프로세스 중에 성공한 리소스 수를 나타냅니다.
- totalCount: 전송 프로세스와 관련된 총 리소스 수를 나타냅니다.

또한 미리 서명된 URL이 포함된 reportUrl 요소가 반환됩니다. 전송 작업에 추가 조사하려는 오류가 있는 경우이 URL을 사용하여 전체 오류 보고서를 다운로드할 수 있습니다.

메타데이터 템플릿 가져오기

단일 대량 가져오기 작업으로 여러 구성 요소, componentTypes 또는 개체를 가져올 수 있습니다. 이 섹션의 예제에서는 이를 수행하는 방법을 보여줍니다.

template: Importing entities

개체를 가져오는 작업에 다음 템플릿 형식을 사용합니다.

```
{
  "entities": [
    {
      "description": "string",
      "entityId": "string",
      "entityName": "string",
      "parentEntityId": "string",
      "tags": {
        "string": "string"
      },
      "components": {
        "string": {
          "componentTypeId": "string",
          "description": "string",
          "properties": {
            "string": {
              "definition": {
                "configuration": {
                  "string": "string"
                },
                "dataType": "DataType",
                "defaultValue": "DataValue",
                "displayName": "string",
                "isExternalId": "boolean",
                "isRequiredInEntity": "boolean",
                "isStoredExternally": "boolean",
                "isTimeSeries": "boolean"
              },
              "value": "DataValue"
            }
          },
```

```
          "propertyGroups": {
            "string": {
              "groupType": "string",
              "propertyNames": [
                "string"
              ]
            }
          }
        }
      }
    }
  ]
}
```

template: Importing componentTypes

componentTypes를 가져오는 작업에 대해 다음 템플릿 형식을 사용합니다.

```
{
  "componentTypes": [
    {
      "componentTypeId": "string",
      "componentTypeName": "string",
      "description": "string",
      "extendsFrom": [
        "string"
      ],
      "functions": {
        "string": {
          "implementedBy": {
            "isNative": "boolean",
            "lambda": {
              "functionName": "Telemetry-tsDataReader",
              "arn": "Telemetry-tsDataReaderARN"
            }
          },
          "requiredProperties": [
            "string"
          ],
          "scope": "string"
        }
      },
      "isSingleton": "boolean",
      "propertyDefinitions": {
```

```
        "string": {
          "configuration": {
            "string": "string"
          },
          "dataType": "DataType",
          "defaultValue": "DataValue",
          "displayName": "string",
          "isExternalId": "boolean",
          "isRequiredInEntity": "boolean",
          "isStoredExternally": "boolean",
          "isTimeSeries": "boolean"
        }
      },
      "propertyGroups": {
        "string": {
          "groupType": "string",
          "propertyNames": [
            "string"
          ]
        }
      },
      "tags": {
        "string": "string"
      }
    }
  ]
}
```

template: Importing components

구성 요소를 가져오는 작업에 다음 템플릿 형식을 사용합니다.

```
{
  "entityComponents": [
    {
      "entityId": "string",
      "componentName": "string",
      "componentTypeId": "string",
      "description": "string",
      "properties": {
        "string": {
          "definition": {
            "configuration": {
              "string": "string"
```

```
            },
            "dataType": "DataType",
            "defaultValue": "DataValue",
            "displayName": "string",
            "isExternalId": "boolean",
            "isRequiredInEntity": "boolean",
            "isStoredExternally": "boolean",
            "isTimeSeries": "boolean"
          },
          "value": "DataValue"
        }
      },
      "propertyGroups": {
        "string": {
          "groupType": "string",
          "propertyNames": [
            "string"
          ]
        }
      }
    }
  ]
}
```

AWS IoT TwinMaker metadataTransferJob 예제

다음 명령을 사용하여 메타데이터 전송을 관리합니다.

- CreateMetadataTransferJob API 작업.

  CLI 명령 예제:

  ```
  aws iottwinmaker create-metadata-transfer-job --region us-east-1 \
  --cli-input-json file://yourTransferFileName.json
  ```

- 작업을 취소하려면 CancelMetadataTransferJob API 작업을 사용합니다.

  CLI 명령 예제:

  ```
  aws iottwinmaker cancel-metadata-transfer-job
  --region us-east-1 \
  ```

```
--metadata-transfer-job-id job-to-cancel-id
```

CancelMetadataTransferJob을 호출하면 특정 메타데이터 전송 작업만 취소되며 이미 내보내거나 가져온 리소스는 영향을 받지 않습니다.

- 특정 작업에 대한 정보를 검색하려면 GetMetadataTransferJob API 작업을 사용합니다.

  CLI 명령 예제:

```
aws iottwinmaker get-metadata-transfer-job \
--metadata-transfer-job-id your_metadata_transfer_job_id \
--region us-east-1 \
```

- 현재 작업을 나열하려면 ListMetadataTransferJobs API 작업을 사용합니다.

  JSON 파일을 사용하여 ListMetadataTransferJobs에서 반환한 결과를 필터링할 수 있습니다. CLI를 사용하여 다음 절차를 참조하세요.

  1. CLI 입력 JSON 파일을 생성하여 사용할 필터를 지정합니다.

```
{
    "sourceType": "s3",
    "destinationType": "iottwinmaker",
    "filters": [{
        "workspaceId": "workspaceforbulkimport"
    },
    {
        "state": "COMPLETED"
    }]
}
```

     파일을 저장하고 파일 이름을 기록합니다. CLI 명령을 입력할 때 필요합니다.

  2. JSON 파일을 다음 CLI 명령의 인수로 사용합니다.

```
aws iottwinmaker list-metadata-transfer-job --region us-east-1 \
--cli-input-json file://ListMetadataTransferJobsExample.json
```

AWS IoT TwinMaker 메타데이터 전송 작업 스키마

metadataTransferJob 가져오기 스키마: Amazon S3 버킷에 업로드할 때이 AWS IoT TwinMaker 메타데이터 스키마를 사용하여 데이터를 검증합니다.

```
{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "title": "IoTTwinMaker",
  "description": "Metadata transfer job resource schema for IoTTwinMaker",
  "definitions": {
    "ExternalId": {
      "type": "string",
      "minLength": 1,
      "maxLength": 128,
      "pattern": "[a-zA-Z0-9][a-zA-Z_\\-0-9.:]*[a-zA-Z0-9]+"
    },
    "Description": {
      "type": "string",
      "minLength": 0,
      "maxLength": 512
    },
    "DescriptionWithDefault": {
      "type": "string",
      "minLength": 0,
      "maxLength": 512,
      "default": ""
    },
    "ComponentTypeName": {
      "description": "A friendly name for the component type.",
      "type": "string",
      "pattern": ".*[^\\u0000-\\u001F\\u007F]*.*",
      "minLength": 1,
      "maxLength": 256
    },
    "ComponentTypeId": {
      "description": "The ID of the component type.",
      "type": "string",
      "pattern": "[a-zA-Z_.\\-0-9:]+",
      "minLength": 1,
      "maxLength": 256
    },
    "ComponentName": {
      "description": "The name of the component.",
```

```
      "type": "string",
      "pattern": "[a-zA-Z_\\-0-9]+",
      "minLength": 1,
      "maxLength": 256
    },
    "EntityId": {
      "description": "The ID of the entity.",
      "type": "string",
      "minLength": 1,
      "maxLength": 128,
      "pattern": "[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}|^[a-zA-
Z0-9][a-zA-Z_\\-0-9.:]*[a-zA-Z0-9]+"
    },
    "EntityName": {
      "description": "The name of the entity.",
      "type": "string",
      "minLength": 1,
      "maxLength": 256,
      "pattern": "[a-zA-Z_0-9-.][a-zA-Z_0-9-. ]*[a-zA-Z0-9]+"
    },
    "ParentEntityId": {
      "description": "The ID of the parent entity.",
      "type": "string",
      "minLength": 1,
      "maxLength": 128,
      "pattern": "\\$ROOT|^[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]
{12}|^[a-zA-Z0-9][a-zA-Z_\\-0-9.:]*[a-zA-Z0-9]+",
      "default": "$ROOT"
    },
    "DisplayName": {
      "description": "A friendly name for the property.",
      "type": "string",
      "pattern": ".*[^\\u0000-\\u001F\\u007F]*.*",
      "minLength": 0,
      "maxLength": 256
    },
    "Tags": {
      "description": "Metadata that you can use to manage the entity / componentType",
      "patternProperties": {
        "^([\\p{L}\\p{Z}\\p{N}_.:/=+\\-@]*)$": {
          "type": "string",
          "minLength": 1,
          "maxLength": 256
        }
```

```
      },
      "existingJavaType": "java.util.Map<String,String>",
      "minProperties": 0,
      "maxProperties": 50
    },
    "Relationship": {
      "description": "The type of the relationship.",
      "type": "object",
      "properties": {
        "relationshipType": {
          "description": "The type of the relationship.",
          "type": "string",
          "pattern": ".*",
          "minLength": 1,
          "maxLength": 256
        },
        "targetComponentTypeId": {
          "description": "The ID of the target component type associated with this
relationship.",
          "$ref": "#/definitions/ComponentTypeId"
        }
      },
      "additionalProperties": false
    },
    "DataValue": {
      "description": "An object that specifies a value for a property.",
      "type": "object",
      "properties": {
        "booleanValue": {
          "description": "A Boolean value.",
          "type": "boolean"
        },
        "doubleValue": {
          "description": "A double value.",
          "type": "number"
        },
        "expression": {
          "description": "An expression that produces the value.",
          "type": "string",
          "pattern": "(^\\$\\{Parameters\\.[a-zA-z]+([a-zA-z_0-9]*)}$)",
          "minLength": 1,
          "maxLength": 316
        },
        "integerValue": {
```

```
          "description": "An integer value.",
          "type": "integer"
        },
        "listValue": {
          "description": "A list of multiple values.",
          "type": "array",
          "minItems": 0,
          "maxItems": 50,
          "uniqueItems": false,
          "insertionOrder": false,
          "items": {
            "$ref": "#/definitions/DataValue"
          },
          "default": null
        },
        "longValue": {
          "description": "A long value.",
          "type": "integer",
          "existingJavaType": "java.lang.Long"
        },
        "stringValue": {
          "description": "A string value.",
          "type": "string",
          "pattern": ".*",
          "minLength": 1,
          "maxLength": 256
        },
        "mapValue": {
          "description": "An object that maps strings to multiple DataValue objects.",
          "type": "object",
          "patternProperties": {
            "[a-zA-Z_\\-0-9]+": {
              "$ref": "#/definitions/DataValue"
            }
          },
          "additionalProperties": {
            "$ref": "#/definitions/DataValue"
          }
        },
        "relationshipValue": {
          "description": "A value that relates a component to another component.",
          "type": "object",
          "properties": {
            "TargetComponentName": {
```

```
              "type": "string",
              "pattern": "[a-zA-Z_\\-0-9]+",
              "minLength": 1,
              "maxLength": 256
            },
            "TargetEntityId": {
              "type": "string",
              "pattern": "[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}|
^[a-zA-Z0-9][a-zA-Z_\\-0-9.:]*[a-zA-Z0-9]+",
              "minLength": 1,
              "maxLength": 128
            }
          },
          "additionalProperties": false
        }
      },
      "additionalProperties": false
    },
    "DataType": {
      "description": "An object that specifies the data type of a property.",
      "type": "object",
      "properties": {
        "allowedValues": {
          "description": "The allowed values for this data type.",
          "type": "array",
          "minItems": 0,
          "maxItems": 50,
          "uniqueItems": false,
          "insertionOrder": false,
          "items": {
            "$ref": "#/definitions/DataValue"
          },
          "default": null
        },
        "nestedType": {
          "description": "The nested type in the data type.",
          "$ref": "#/definitions/DataType"
        },
        "relationship": {
          "description": "A relationship that associates a component with another
 component.",
          "$ref": "#/definitions/Relationship"
        },
        "type": {
```

```
          "description": "The underlying type of the data type.",
          "type": "string",
          "enum": [
            "RELATIONSHIP",
            "STRING",
            "LONG",
            "BOOLEAN",
            "INTEGER",
            "DOUBLE",
            "LIST",
            "MAP"
          ]
        },
        "unitOfMeasure": {
          "description": "The unit of measure used in this data type.",
          "type": "string",
          "pattern": ".*",
          "minLength": 1,
          "maxLength": 256
        }
      },
      "required": [
        "type"
      ],
      "additionalProperties": false
    },
    "PropertyDefinition": {
      "description": "An object that specifies information about a property.",
      "type": "object",
      "properties": {
        "configuration": {
          "description": "An object that specifies information about a property.",
          "patternProperties": {
            "[a-zA-Z_\\-0-9]+": {
              "type": "string",
              "pattern": "[a-zA-Z_\\-0-9]+",
              "minLength": 1,
              "maxLength": 256
            }
          },
          "existingJavaType": "java.util.Map<String,String>"
        },
        "dataType": {
          "description": "An object that contains information about the data type.",
```

```
          "$ref": "#/definitions/DataType"
        },
        "defaultValue": {
          "description": "An object that contains the default value.",
          "$ref": "#/definitions/DataValue"
        },
        "displayName": {
          "description": "An object that contains the default value.",
          "$ref": "#/definitions/DisplayName"
        },
        "isExternalId": {
          "description": "A Boolean value that specifies whether the property ID comes
from an external data store.",
          "type": "boolean",
          "default": null
        },
        "isRequiredInEntity": {
          "description": "A Boolean value that specifies whether the property is
required.",
          "type": "boolean",
          "default": null
        },
        "isStoredExternally": {
          "description": "A Boolean value that specifies whether the property is stored
externally.",
          "type": "boolean",
          "default": null
        },
        "isTimeSeries": {
          "description": "A Boolean value that specifies whether the property consists
of time series data.",
          "type": "boolean",
          "default": null
        }
      },
      "additionalProperties": false
    },
    "PropertyDefinitions": {
      "type": "object",
      "patternProperties": {
        "[a-zA-Z_\\-0-9]+": {
          "$ref": "#/definitions/PropertyDefinition"
        }
      },
```

```
      "additionalProperties": {
        "$ref": "#/definitions/PropertyDefinition"
      }
    },
    "Property": {
      "type": "object",
      "properties": {
        "definition": {
          "description": "The definition of the property",
          "$ref": "#/definitions/PropertyDefinition"
        },
        "value": {
          "description": "The value of the property.",
          "$ref": "#/definitions/DataValue"
        }
      },
      "additionalProperties": false
    },
    "Properties": {
      "type": "object",
      "patternProperties": {
        "[a-zA-Z_\\-0-9]+": {
          "$ref": "#/definitions/Property"
        }
      },
      "additionalProperties": {
        "$ref": "#/definitions/Property"
      }
    },
    "PropertyName": {
      "type": "string",
      "pattern": "[a-zA-Z_\\-0-9]+"
    },
    "PropertyGroup": {
      "description": "An object that specifies information about a property group.",
      "type": "object",
      "properties": {
        "groupType": {
          "description": "The type of property group.",
          "type": "string",
          "enum": [
            "TABULAR"
          ]
        },
```

```
        "propertyNames": {
          "description": "The list of property names in the property group.",
          "type": "array",
          "minItems": 1,
          "maxItems": 256,
          "uniqueItems": true,
          "insertionOrder": false,
          "items": {
            "$ref": "#/definitions/PropertyName"
          },
          "default": null
        }
      },
      "additionalProperties": false
    },
    "PropertyGroups": {
      "type": "object",
      "patternProperties": {
        "[a-zA-Z_\\-0-9]+": {
          "$ref": "#/definitions/PropertyGroup"
        }
      },
      "additionalProperties": {
        "$ref": "#/definitions/PropertyGroup"
      }
    },
    "Component": {
      "type": "object",
      "properties": {
        "componentTypeId": {
          "$ref": "#/definitions/ComponentTypeId"
        },
        "description": {
          "$ref": "#/definitions/Description"
        },
        "properties": {
          "description": "An object that maps strings to the properties to set in the
component type. Each string in the mapping must be unique to this object.",
          "$ref": "#/definitions/Properties"
        },
        "propertyGroups": {
          "description": "An object that maps strings to the property groups to set in
the entity component. Each string in the mapping must be unique to this object.",
          "$ref": "#/definitions/PropertyGroups"
```

```
        }
      },
      "required": [
        "componentTypeId"
      ],
      "additionalProperties": false
    },
    "RequiredProperty": {
      "type": "string",
      "pattern": "[a-zA-Z_\\-0-9]+"
    },
    "LambdaFunction": {
      "type": "object",
      "properties": {
        "arn": {
          "type": "string",
          "pattern": "arn:((aws)|(aws-cn)|(aws-us-gov)|(\\${partition})):lambda:(([a-
z0-9-]+)|(\\${region})):([0-9]{12}|(\\${accountId})):function:[/a-zA-Z0-9_-]+",
          "minLength": 1,
          "maxLength": 128
        }
      },
      "additionalProperties": false,
      "required": [
        "arn"
      ]
    },
    "DataConnector": {
      "description": "The data connector.",
      "type": "object",
      "properties": {
        "isNative": {
          "description": "A Boolean value that specifies whether the data connector is
 native to IoT TwinMaker.",
          "type": "boolean"
        },
        "lambda": {
          "description": "The Lambda function associated with this data connector.",
          "$ref": "#/definitions/LambdaFunction"
        }
      },
      "additionalProperties": false
    },
    "Function": {
```

```
      "description": "The function of component type.",
      "type": "object",
      "properties": {
        "implementedBy": {
          "description": "The data connector.",
          "$ref": "#/definitions/DataConnector"
        },
        "requiredProperties": {
          "description": "The required properties of the function.",
          "type": "array",
          "minItems": 1,
          "maxItems": 256,
          "uniqueItems": true,
          "insertionOrder": false,
          "items": {
            "$ref": "#/definitions/RequiredProperty"
          },
          "default": null
        },
        "scope": {
          "description": "The scope of the function.",
          "type": "string",
          "enum": [
            "ENTITY",
            "WORKSPACE"
          ]
        }
      },
      "additionalProperties": false
    },
    "Entity": {
      "type": "object",
      "properties": {
        "description": {
          "description": "The description of the entity.",
          "$ref": "#/definitions/DescriptionWithDefault"
        },
        "entityId": {
          "$ref": "#/definitions/EntityId"
        },
        "entityExternalId": {
          "description": "The external ID of the entity.",
          "$ref": "#/definitions/ExternalId"
        },
```

```
        "entityName": {
          "$ref": "#/definitions/EntityName"
        },
        "parentEntityId": {
          "$ref": "#/definitions/ParentEntityId"
        },
        "tags": {
          "$ref": "#/definitions/Tags"
        },
        "components": {
          "description": "A map that sets information about a component.",
          "type": "object",
          "patternProperties": {
            "[a-zA-Z_\\-0-9]+": {
              "$ref": "#/definitions/Component"
            }
          },
          "additionalProperties": {
            "$ref": "#/definitions/Component"
          }
        }
      },
      "required": [
        "entityId",
        "entityName"
      ],
      "additionalProperties": false
    },
    "ComponentType": {
      "type": "object",
      "properties": {
        "description": {
          "description": "The description of the component type.",
          "$ref": "#/definitions/DescriptionWithDefault"
        },
        "componentTypeId": {
          "$ref": "#/definitions/ComponentTypeId"
        },
        "componentTypeExternalId": {
          "description": "The external ID of the component type.",
          "$ref": "#/definitions/ExternalId"
        },
        "componentTypeName": {
          "$ref": "#/definitions/ComponentTypeName"
```

```
        },
        "extendsFrom": {
          "description": "Specifies the parent component type to extend.",
          "type": "array",
          "minItems": 1,
          "maxItems": 256,
          "uniqueItems": true,
          "insertionOrder": false,
          "items": {
            "$ref": "#/definitions/ComponentTypeId"
          },
          "default": null
        },
        "functions": {
          "description": "a Map of functions in the component type. Each function's key
must be unique to this map.",
          "type": "object",
          "patternProperties": {
            "[a-zA-Z_\\-0-9]+": {
              "$ref": "#/definitions/Function"
            }
          },
          "additionalProperties": {
            "$ref": "#/definitions/Function"
          }
        },
        "isSingleton": {
          "description": "A Boolean value that specifies whether an entity can have
more than one component of this type.",
          "type": "boolean",
          "default": false
        },
        "propertyDefinitions": {
          "description": "An map of the property definitions in the component type.
Each property definition's key must be unique to this map.",
          "$ref": "#/definitions/PropertyDefinitions"
        },
        "propertyGroups": {
          "description": "An object that maps strings to the property groups to set in
the component type. Each string in the mapping must be unique to this object.",
          "$ref": "#/definitions/PropertyGroups"
        },
        "tags": {
          "$ref": "#/definitions/Tags"
```

```
        }
      },
      "required": [
        "componentTypeId"
      ],
      "additionalProperties": false
    },
    "EntityComponent": {
      "type": "object",
      "properties": {
        "entityId": {
          "$ref": "#/definitions/EntityId"
        },
        "componentName": {
          "$ref": "#/definitions/ComponentName"
        },
        "componentExternalId": {
          "description": "The external ID of the component.",
          "$ref": "#/definitions/ExternalId"
        },
        "componentTypeId": {
          "$ref": "#/definitions/ComponentTypeId"
        },
        "description": {
          "description": "The description of the component.",
          "$ref": "#/definitions/Description"
        },
        "properties": {
          "description": "An object that maps strings to the properties to set in the
component. Each string in the mapping must be unique to this object.",
          "$ref": "#/definitions/Properties"
        },
        "propertyGroups": {
          "description": "An object that maps strings to the property groups to set in
the component. Each string in the mapping must be unique to this object.",
          "$ref": "#/definitions/PropertyGroups"
        }
      },
      "required": [
        "entityId",
        "componentTypeId",
        "componentName"
      ],
      "additionalProperties": false
```

```
    }
  },
  "additionalProperties": false,
  "properties": {
    "entities": {
      "type": "array",
      "uniqueItems": false,
      "items": {
        "$ref": "#/definitions/Entity"
      }
    },
    "componentTypes": {
      "type": "array",
      "uniqueItems": false,
      "items": {
        "$ref": "#/definitions/ComponentType"
      }
    },
    "entityComponents": {
      "type": "array",
      "uniqueItems": false,
      "items": {
        "$ref": "#/definitions/EntityComponent"
      },
      "default": null
    }
  }
}
```

다음은 라는 새 componentType을 `component.type.intial` 생성하고 라는 엔터티를 생성하는 예제입니다`initial`.

```
{
  "componentTypes": [
    {
      "componentTypeId": "component.type.initial",
      "tags": {
        "key": "value"
      }
    }
  ],
  "entities": [
    {
```

```
      "entityName": "initial",
      "entityId": "initial"
    }
  ]
}
```

다음은 기존 엔터티를 업데이트하는 예입니다.

```
{
  "componentTypes": [
    {
      "componentTypeId": "component.type.initial",
      "description": "updated"
    }
  ],
  "entities": [
    {
      "entityName": "parent",
      "entityId": "parent"
    },
    {
      "entityName": "child",
      "entityId": "child",
      "components": {
        "testComponent": {
          "componentTypeId": "component.type.initial",
          "properties": {
            "testProperty": {
              "definition": {
                "configuration": {
                  "alias": "property"
                },
                "dataType": {
                  "relationship": {
                    "relationshipType": "parent",
                    "targetComponentTypeId": "test"
                  },
                  "type": "STRING",
                  "unitOfMeasure": "t"
                },
                "displayName": "displayName"
              }
            }
```

```
          }
        }
      },
      "parentEntityId": "parent"
    }
  ],
  "entityComponents": [
    {
      "entityId": "initial",
      "componentTypeId": "component.type.initial",
      "componentName": "entityComponent",
      "description": "additionalDescription",
      "properties": {
        "additionalProperty": {
          "definition": {
            "configuration": {
              "alias": "additionalProperty"
            },
            "dataType": {
              "type": "STRING"
            },
            "displayName": "additionalDisplayName"
          },
          "value": {
            "stringValue": "test"
          }
        }
      }
    }
  ]
}
```

AWS IoT TwinMaker 데이터 커넥터

AWS IoT TwinMaker 는 커넥터 기반 아키텍처를 사용하므로 자체 데이터 스토어의 데이터를에 연결할 수 있습니다 AWS IoT TwinMaker. 즉,를 사용하기 전에 데이터를 마이그레이션할 필요가 없습니다 AWS IoT TwinMaker. 현재는에 대한 자사 커넥터를 AWS IoT TwinMaker 지원합니다 AWS IoT SiteWise. 모델링 및 속성 데이터를에 저장하는 AWS IoT SiteWise경우 자체 커넥터를 구현할 필요가 없습니다. Timestream, DynamoDB 또는 Snowflake와 같은 다른 데이터 스토어에 모델링 또는 속성 데이터를 저장하는 경우 필요한 경우가 AWS Lambda 커넥터를 호출할 AWS IoT TwinMaker 수 있도록 AWS IoT TwinMaker 데이터 커넥터 인터페이스를 사용하여 커넥터를 구현해야 합니다.

주제

AWS IoT TwinMaker 데이터 커넥터

전송된 쿼리를 해결하고 결과 또는 오류를 반환하려면 커넥터가 기본 데이터 저장소에 액세스할 수 있어야 합니다.

사용 가능한 커넥터, 요청 인터페이스 및 응답 인터페이스에 대해 알아보려면 다음 주제를 참조하십시오.

커넥터 인터페이스에서 사용되는 속성에 대한 자세한 내용은 GetPropertyValueHistory API 작업을 참조하십시오.

> **Note**
>
> 일부 커넥터에는 요청 및 응답 인터페이스 모두에 시작 시간 및 종료 시간 속성에 대한 타임스탬프 필드가 두 개 있습니다. `startDateTime` 및 `endDateTime` 모두 에포크 세컨드를 나타내는 데 긴 숫자를 사용하지만, 더 이상 지원되지 않습니다. 이전 버전과의 호환성을 유지하기 위해 여전히 해당 필드에 타임스탬프 값을 전송하지만 API 타임스탬프 형식과 일치하는 `startTime` 및 `endTime` 필드를 사용하는 것이 좋습니다.

주제

스키마 이니셜라이저 커넥터

구성 요소 유형 또는 개체 수명 주기의 스키마 이니셜라이저를 사용하여 기본 데이터 소스에서 구성 요소 유형 또는 구성 요소 속성을 가져올 수 있습니다. 스키마 이니셜라이저는 API 작업을 명시적으로 직접적으로 호출하여 `properties`을 설정하지 않고도 구성 요소 유형이나 구성 요소 속성을 자동으로 가져옵니다.

스키마이니셜라이저 요청 인터페이스

```
{
  "workspaceId": "string",
  "entityId": "string",
  "componentName": "string",
  "properties": {
    // property name as key,
    // value is of type PropertyRequest
    "string": "PropertyRequest"
  }
}
```

> **Note**
>
> 이 요청 인터페이스의 속성 맵은 `PropertyRequest`입니다. 자세한 내용은 PropertyRequest를 참조하십시오.

스키마 이니셜라이저 응답 인터페이스

```
{
```

```
  "properties": {
    // property name as key,
    // value is of type PropertyResponse
    "string": "PropertyResponse"
  }
}
```

> **Note**
>
> 이 요청 인터페이스의 속성 맵은 PropertyResponse입니다. 자세한 내용은 PropertyResponse를 참조하십시오.

DataReaderByEntity

DataReaderByEntity는 단일 구성 요소에 있는 속성의 시계열 값을 가져오는 데 사용되는 데이터 영역 커넥터입니다.

이 커넥터의 속성 유형, 구문 및 형식에 대한 자세한 내용은 GetPropertyValueHistory API 작업을 참조하십시오.

DataReaderByEntity 요청 인터페이스

```
{
  "startDateTime": long, // In epoch sec, deprecated
  "startTime": "string", // ISO-8601 timestamp format
  "endDateTime": long, // In epoch sec, deprecated
  "endTime": "string", // ISO-8601 timestamp format
  "properties": {
    // A map of properties as in the get-entity API response
    // property name as key,
    // value is of type PropertyResponse
    "string": "PropertyResponse"
   },
  "workspaceId": "string",
  "selectedProperties": List:"string",
  "propertyFilters": List:PropertyFilter,
  "entityId": "string",
  "componentName": "string",
  "componentTypeId": "string",
  "interpolation": InterpolationParameters,
  "nextToken": "string",
```

```
  "maxResults": int,
  "orderByTime": "string"
  }
```

DataReaderByEntity 응답 인터페이스

```
{
  "propertyValues": [
    {
      "entityPropertyReference": EntityPropertyReference, // The same
 as EntityPropertyReference
      "values": [
        {
        "timestamp": long, // Epoch sec, deprecated
        "time": "string", // ISO-8601 timestamp format
        "value": DataValue // The same as DataValue
        }
      ]
    }
  ],
  "nextToken": "string"
}
```

DataReaderByComponentType

동일한 구성 요소 유형에서 오는 공통 속성의 시계열 값을 가져오려면 데이터 영역 커넥터 DataReaderByEntity를 사용하십시오. 예를 들어 구성 요소 유형에 시계열 속성을 정의하고 해당 구성 요소 유형을 사용하는 구성 요소가 여러 개 있는 경우, 지정된 시간 범위의 모든 구성 요소에서 해당 속성을 쿼리할 수 있습니다. 이를 위한 일반적인 사용 사례는 여러 구성 요소의 경보 상태를 쿼리하여 개체를 전체적으로 파악하려는 경우입니다.

이 커넥터의 속성 유형, 구문 및 형식에 대한 자세한 내용은 GetPropertyValueHistory API 작업을 참조하십시오.

DataReaderByComponentType 요청 인터페이스

```
{
  "startDateTime": long, // In epoch sec, deprecated
  "startTime": "string", // ISO-8601 timestamp format
  "endDateTime": long, // In epoch sec, deprecated
  "endTime": "string", // ISO-8601 timestamp format
  "properties": { // A map of properties as in the get-entity API response
```

```
    // property name as key,
    // value is of type PropertyResponse
    "string": "PropertyResponse"
   },
  "workspaceId": "string",
  "selectedProperties": List:"string",
  "propertyFilters": List:PropertyFilter,
  "componentTypeId": "string",
  "interpolation": InterpolationParameters,
  "nextToken": "string",
  "maxResults": int,
  "orderByTime": "string"
}
```

DataReaderByComponentType 응답 인터페이스

```
{
  "propertyValues": [
    {
      "entityPropertyReference": EntityPropertyReference, // The same
 as EntityPropertyReference
      "entityId": "string",
      "componentName": "string",
      "values": [
        {
        "timestamp": long, // Epoch sec, deprecated
        "time": "string", // ISO-8601 timestamp format
        "value": DataValue // The same as DataValue
        }
      ]
    }
  ],
  "nextToken": "string"
}
```

DataReader

Data Reader는 DataReaderByEntity 및DataReaderByComponentType 사례 모두를 처리할 수 있는 데이터 영역 커넥터입니다.

이 커넥터의 속성 유형, 구문 및 형식에 대한 자세한 내용은 GetPropertyValueHistory API 작업을 참조하십시오.

DataReader 요청 인터페이스

`EntityId` 및 `componentName`는 선택 사항입니다.

```
{
  "startDateTime": long, // In epoch sec, deprecated
  "startTime": "string", // ISO-8601 timestamp format
  "endDateTime": long, // In epoch sec, deprecated
  "endTime": "string", // ISO-8601 timestamp format
  "properties": { // A map of properties as in the get-entity API response
    // property name as key,
    // value is of type PropertyRequest
    "string": "PropertyRequest"
  },

  "workspaceId": "string",
  "selectedProperties": List:"string",
  "propertyFilters": List:PropertyFilter,
  "entityId": "string",
  "componentName": "string",
  "componentTypeId": "string",
  "interpolation": InterpolationParameters,
  "nextToken": "string",
  "maxResults": int,
  "orderByTime": "string"
}
```

DataReader 응답 인터페이스

```
{
  "propertyValues": [
    {
      "entityPropertyReference": EntityPropertyReference, // The same
 as EntityPropertyReference
      "values": [
        {
        "timestamp": long, // Epoch sec, deprecated
        "time": "string", // ISO-8601 timestamp format
        "value": DataValue // The same as DataValue
        }
      ]
    }
  ],
```

```
  "nextToken": "string"
}
```

AttributePropertyValueReaderByEntity

AttributePropertyValueReader ByEntity는 단일 개체의 정적 속성 값을 가져오는 데 사용할 수 있는 데이터 영역 커넥터입니다.

이 커넥터의 속성 유형, 구문 및 형식에 대한 자세한 내용은 GetPropertyValue API 작업을 참조하십시오.

AttributePropertyValueReaderByEntity 요청 인터페이스

```
{
  "properties": {
    // property name as key,
    // value is of type PropertyResponse
    "string": "PropertyResponse"
  }

  "workspaceId": "string",
  "entityId": "string",
  "componentName": "string",
  "selectedProperties": List:"string",
}
```

AttributePropertyValueReaderByEntity 응답 인터페이스

```
{
  "propertyValues": {
    "string": { // property name as key
        "propertyReference": EntityPropertyReference, // The same
 as EntityPropertyReference
        "propertyValue": DataValue // The same as DataValue
    }
}
```

DataWriter

DataWriter는 단일 구성요소의 속성을 위해 시계열 데이터 포인트를 기본 데이터 저장소에 다시 쓰는 데 사용할 수 있는 데이터 영역 커넥터입니다.

이 커넥터의 속성 유형, 구문 및 형식에 대한 자세한 내용은 BatchPutPropertyValues API 작업을 참조하십시오.

DataWriter 요청 인터페이스

```
{
  "workspaceId": "string",
  "properties": {
    // entity id as key
    "String": {
      // property name as key,
      // value is of type PropertyResponse
      "string": PropertyResponse
    }
  },
  "entries": [
    {
      "entryId": "string",
      "entityPropertyReference": EntityPropertyReference, // The same
 as EntityPropertyReference
      "propertyValues": [
        {
        "timestamp": long, // Epoch sec, deprecated
        "time": "string", // ISO-8601 timestamp format
        "value": DataValue // The same as DataValue
        }
      ]
    }
  ]
}
```

DataWriter 응답 인터페이스

```
{
  "errorEntries": [
    {
      "errors": List:BatchPutPropertyError // The value is a list of
 type BatchPutPropertyError
    }
  ]
}
```

예제

다음 JSON 샘플은 여러 커넥터에 대한 응답 및 요청 구문의 예입니다.

- SchemaInitializer:

  다음 예제는 구성 요소 유형 수명 주기의 스키마 이니셜라이저를 보여줍니다.

  요청:

```
{
  "workspaceId": "myWorkspace",
  "properties": {
    "modelId": {
      "definition": {
          "dataType": { "type": "STRING" },
          "isExternalId": true,
          "isFinal": true,
          "isImported": false,
          "isInherited": false,
          "isRequiredInEntity": true,
          "isStoredExternally": false,
          "isTimeSeries": false,
          "defaultValue": {
              "stringValue": "myModelId"
          }
      },
      "value": {
          "stringValue": "myModelId"
      }
    },
    "tableName": {
      "definition": {
          "dataType": { "type": "STRING" },
          "isExternalId": false,
          "isFinal": false,
          "isImported": false,
          "isInherited": false,
          "isRequiredInEntity": false,
          "isStoredExternally": false,
          "isTimeSeries": false,
          "defaultValue": {
              "stringValue": "myTableName"
```

```
            }
        },
        "value": {
            "stringValue": "myTableName"
        }
    }
  }
}
```

응답:

```
{
  "properties": {
    "myProperty1": {
      "definition": {
        "dataType": {
          "type": "DOUBLE",
          "unitOfMeasure": "%"
        },
        "configuration": {
          "myProperty1Id": "idValue"
        },
        "isTimeSeries": true
      }
    },
    "myProperty2": {
      "definition": {
        "dataType": { "type": "STRING" },
        "isTimeSeries": false,
        "defaultValue": {
          "stringValue": "property2Value"
        }
      }
    }
  }
}
```

- 개체 수명 주기의 스키마 이니셜라이저:

  요청:

```
{
  "workspaceId": "myWorkspace",
```

```
  "entityId": "myEntity",
  "componentName": "myComponent",
  "properties": {
    "assetId": {
      "definition": {
          "dataType": { "type": "STRING" },
          "isExternalId": true,
          "isFinal": true,
          "isImported": false,
          "isInherited": false,
          "isRequiredInEntity": true,
          "isStoredExternally": false,
          "isTimeSeries": false
      },
      "value": {
          "stringValue": "myAssetId"
      }
    },
    "tableName": {
      "definition": {
          "dataType": { "type": "STRING" },
          "isExternalId": false,
          "isFinal": false,
          "isImported": false,
          "isInherited": false,
          "isRequiredInEntity": false,
          "isStoredExternally": false,
          "isTimeSeries": false
      },
      "value": {
          "stringValue": "myTableName"
      }
    }
  }
}
```

응답:

```
{
  "properties": {
    "myProperty1": {
      "definition": {
        "dataType": {
```

```
          "type": "DOUBLE",
          "unitOfMeasure": "%"
        },
        "configuration": {
          "myProperty1Id": "idValue"
        },
        "isTimeSeries": true
      }
    },
    "myProperty2": {
      "definition": {
        "dataType": { "type": "STRING" },
        "isTimeSeries": false
      },
      "value": {
        "stringValue": "property2Value"
      }
    }
  }
}
```

- DataReaderByEntity and DataReader:

  요청:

```
{
  "workspaceId": "myWorkspace",
  "entityId": "myEntity",
  "componentName": "myComponent",
  "selectedProperties": [
    "Temperature",
    "Pressure"
  ],
  "startTime": "2022-04-07T04:04:42Z",
  "endTime": "2022-04-07T04:04:45Z",
  "maxResults": 4,
  "orderByTime": "ASCENDING",
  "properties": {
      "assetId": {
          "definition": {
              "dataType": { "type": "STRING" },
              "isExternalId": true,
              "isFinal": true,
              "isImported": false,
```

```
                "isInherited": false,
                "isRequiredInEntity": true,
                "isStoredExternally": false,
                "isTimeSeries": false
            },
            "value": {
                "stringValue": "myAssetId"
            }
        },
        "Temperature": {
            "definition": {
                "configuration": {
                    "temperatureId": "xyz123"
                },
                "dataType": {
                    "type": "DOUBLE",
                    "unitOfMeasure": "DEGC"
                },
                "isExternalId": false,
                "isFinal": false,
                "isImported": true,
                "isInherited": false,
                "isRequiredInEntity": false,
                "isStoredExternally": false,
                "isTimeSeries": true
            }
        },
        "Pressure": {
            "definition": {
                "configuration": {
                    "pressureId": "xyz456"
                },
                "dataType": {
                    "type": "DOUBLE",
                    "unitOfMeasure": "MPA"
                },
                "isExternalId": false,
                "isFinal": false,
                "isImported": true,
                "isInherited": false,
                "isRequiredInEntity": false,
                "isStoredExternally": false,
                "isTimeSeries": true
            }
```

```
      }
  }
}
```

응답:

```
{
  "propertyValues": [
    {
      "entityPropertyReference": {
        "entityId": "myEntity",
        "componentName": "myComponent",
        "propertyName": "Temperature"
      },
      "values": [
        {
          "time": "2022-04-07T04:04:42Z",
          "value": {
            "doubleValue": 588.168
          }
        },
        {
          "time": "2022-04-07T04:04:43Z",
          "value": {
            "doubleValue": 592.4224
          }
        }
      ]
    }
  ],
  "nextToken": "qwertyuiop"
}
```

- AttributePropertyValueReaderByEntity:

  요청:

```
{
  "workspaceId": "myWorkspace",
  "entityId": "myEntity",
  "componentName": "myComponent",
  "selectedProperties": [
    "manufacturer",
```

```
  ],
  "properties": {
    "assetId": {
      "definition": {
        "dataType": { "type": "STRING" },
        "isExternalId": true,
        "isFinal": true,
        "isImported": false,
        "isInherited": false,
        "isRequiredInEntity": true,
        "isStoredExternally": false,
        "isTimeSeries": false
      },
      "value": {
          "stringValue": "myAssetId"
      }
    },
    "manufacturer": {
      "definition": {
        "dataType": { "type": "STRING" },
        "configuration": {
            "manufacturerPropId": "M001"
        },
        "isExternalId": false,
        "isFinal": false,
        "isImported": false,
        "isInherited": false,
        "isRequiredInEntity": false,
        "isStoredExternally": true,
        "isTimeSeries": false
      }
    }
  }
}
```

응답:

```
{
  "propertyValues": {
    "manufacturer": {
      "propertyReference": {
        "propertyName": "manufacturer",
```

```
        "entityId": "myEntity",
        "componentName": "myComponent"
      },
      "propertyValue": {
        "stringValue": "Amazon"
      }
    }
  }
}
```

- DataWriter:

  요청:

```
{
  "workspaceId": "myWorkspaceId",
  "properties": {
    "myEntity": {
      "Temperature": {
          "definition": {
              "configuration": {
                  "temperatureId": "xyz123"
              },
              "dataType": {
                  "type": "DOUBLE",
                  "unitOfMeasure": "DEGC"
              },
              "isExternalId": false,
              "isFinal": false,
              "isImported": true,
              "isInherited": false,
              "isRequiredInEntity": false,
              "isStoredExternally": false,
              "isTimeSeries": true
          }
      }
    }
  },
  "entries": [
    {
      "entryId": "myEntity",
      "entityPropertyReference": {
        "entityId": "myEntity",
        "componentName": "myComponent",
```

```
        "propertyName": "Temperature"
      },
      "propertyValues": [
        {
          "timestamp": 1626201120,
          "value": {
            "doubleValue": 95.6958
          }
        },
        {
          "timestamp": 1626201132,
          "value": {
            "doubleValue": 80.6959
          }
        }
      ]
    }
  ]
}
```

응답:

```
{
    "errorEntries": [
        {
            "errors": [
                {
                    "errorCode": "409",
                    "errorMessage": "Conflict value at same timestamp",
                    "entry": {
                        "entryId": "myEntity",
                        "entityPropertyReference": {
                            "entityId": "myEntity",
                            "componentName": "myComponent",
                            "propertyName": "Temperature"
                        },
                        "propertyValues": [
                            "time": "2022-04-07T04:04:42Z",
                            "value": {
                                "doubleValue": 95.6958
                            }
                        ]
```

```
                    }
                }
            ]
        }
    ]
}
```

AWS IoT TwinMaker Athena 테이블 형식 데이터 커넥터

Athena 테이블 형식 데이터 커넥터를 사용하면 AWS IoT TwinMaker에서 Athena 데이터 저장소에 액세스하여 사용할 수 있습니다. 집중적인 데이터 마이그레이션 작업 없이 Athena 데이터를 사용하여 디지털 트윈을 구축할 수 있습니다. 사전 구축된 커넥터를 사용하거나 사용자 지정 Athena 커넥터를 생성하여 Athena 데이터 소스의 데이터에 액세스할 수 있습니다.

AWS IoT TwinMaker Athena 데이터 커넥터 사전 조건

Athena 테이블 형식 데이터 커넥터를 사용하기 전에 다음 사전 조건을 완료합니다.

- 관리형 Athena 테이블 및 관련 Amazon S3 리소스를 생성합니다. Athena 사용에 대한 자세한 내용은 Athena 설명서를 참조하십시오.
- AWS IoT TwinMaker 워크스페이스를 생성합니다. AWS IoT TwinMaker 콘솔에서 작업 영역을 생성할 수 있습니다.
- Athena 권한으로 워크스페이스 IAM 역할을 업데이트합니다. 자세한 내용은 Athena 데이터 커넥터를 사용하도록 워크스페이스 IAM 역할을 수정합니다. 단원을 참조하십시오.
- AWS IoT TwinMaker의 개체 구성 요소 시스템과 개체를 생성하는 방법을 숙지합니다. 자세한 내용은 첫 번째 개체 생성 단원을 참조하십시오.
- AWS IoT TwinMaker의 데이터 커넥터에 익숙해지세요. 자세한 내용은 AWS IoT TwinMaker 데이터 커넥터 단원을 참조하십시오.

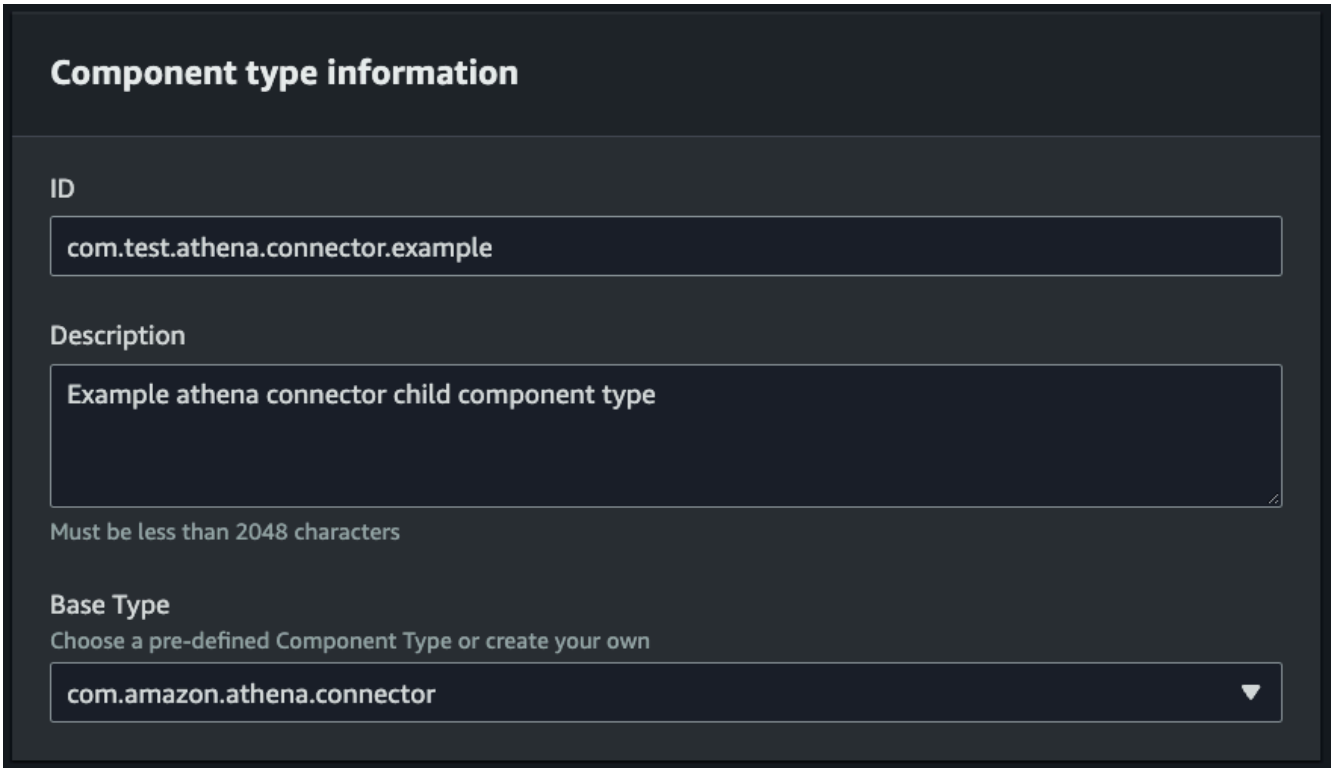
Athena 데이터 커넥터 사용하기

Athena 데이터 커넥터를 사용하려면 Athena 커넥터를 구성 요소 유형으로 사용하여 구성 요소를 만들어야 합니다. 그런 다음 구성 요소를 장면 내의 개체에 첨부하여 AWS IoT TwinMaker에서 사용할 수 있습니다.

Athena 데이터 커넥터를 사용하여 구성 요소 유형 만들기

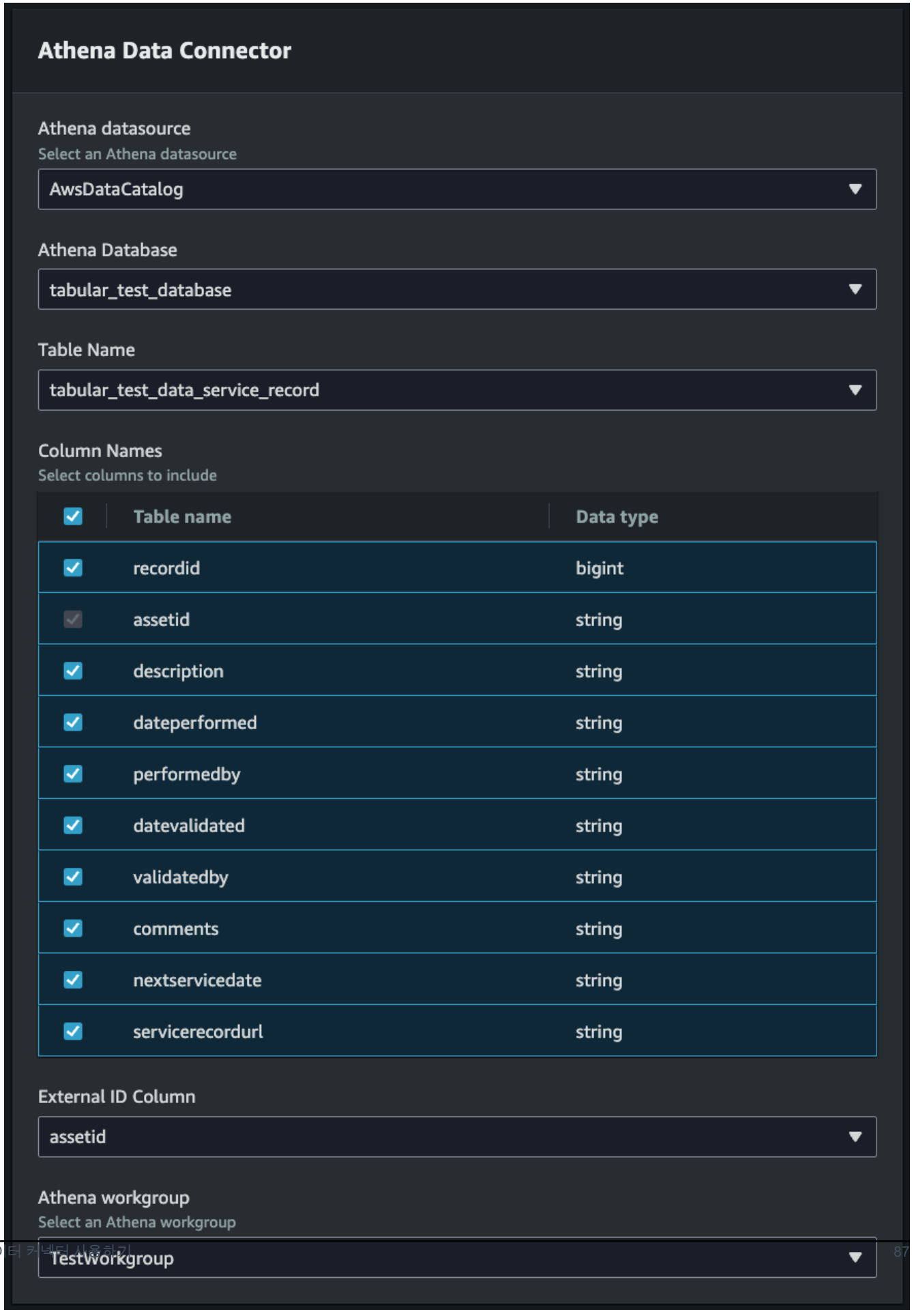
다음 절차에 따라 Athena 테이블 형식 데이터 커넥터를 사용하여 AWS IoT TwinMaker 구성 요소 유형을 생성합니다.

1. AWS IoT TwinMaker 콘솔로 이동합니다.
2. 기존 작업 영역을 열거나 새 작업 영역을 생성합니다.
3. 왼쪽 탐색 메뉴에서 구성 요소 유형을 선택하고 구성 요소 유형 생성을 선택하여 구성요소 유형 생성 페이지를 엽니다.
4. 구성 요소 유형 생성 페이지에서 사용 사례에 맞는 ID로 ID 필드를 채웁니다.



5. 기본 유형을 선택합니다. 드롭다운 목록에서 com.amazon.athena.connector라는 레이블이 붙은 Athena 테이블 형식 데이터 커넥터를 선택합니다.
6. 다음 필드에서 Athena 리소스를 선택하여 구성 요소 유형의 데이터 소스를 구성합니다.
   - Athena 데이터 소스를 선택합니다.
   - Athena database 데이터베이스를 선택합니다.
   - Table name을 선택합니다.
   - Athena workGroup을 선택하십시오.
7. 데이터 소스로 사용할 Athena 리소스를 선택한 후 테이블에서 포함할 열을 선택합니다.

8. External ID column name을 선택합니다. 이전 단계에서 외부 ID 열로 사용할 열을 선택합니다. 외부 ID는 Athena 자산을 나타내고 AWS IoT TwinMaker 개체에 매핑하는 데 사용되는 ID입니다.


Athena Data Connector
Athena datasource
Select an Athena datasource
AwsDataCatalog
Athena Database
tabular_test_database
Table Name
tabular_test_data_service_record
Column Names
Select columns to include
Table name
Data type
recordid
bigint
assetid
string
description
string
dateperformed
string
performedby
string
datevalidated
string
validatedby
string
comments
string
nextservicedate
string
servicerecordurl
string
External ID Column
assetid
Athena workgroup
Select an Athena workgroup
TestWorkgroup

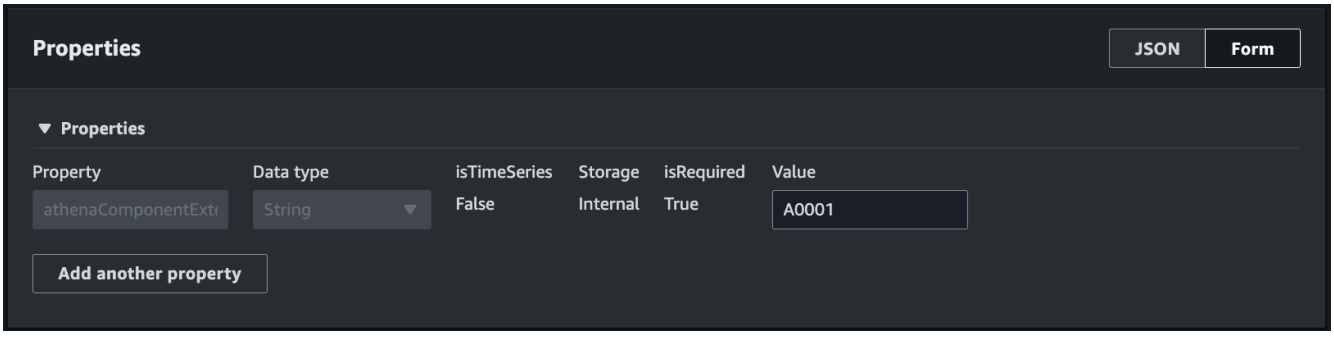
9. (선택 사항) 이러한 리소스에 AWS 태그를 추가하여 태그를 그룹화하고 구성할 수 있습니다.
10. 구성 요소 유형 생성을 선택하여 구성 요소 유형 생성을 완료합니다.

Athena 데이터 커넥터 유형으로 컴포넌트를 생성하여 개체에 연결

다음 절차에 따라 Athena 테이블 형식 데이터 커넥터를 사용하여 AWS IoT TwinMaker 구성 요소를 생성하고 개체에 연결합니다.

> **Note**
>
> 이 절차를 완료하려면 Athena 테이블 형식 데이터 커넥터를 데이터 소스로 사용하는 기존 구성 요소 유형이 있어야 합니다. 이 연습을 시작하기 전에 이전의 절차인 Athena 데이터 커넥터를 사용하여 구성 요소 유형 만들기를 참조하십시오.

1. AWS IoT TwinMaker 콘솔로 이동합니다.
2. 기존 작업 영역을 열거나 새 작업 영역을 생성합니다.
3. 왼쪽 탐색 메뉴에서 개체를 선택하고 구성 요소를 추가하거나 새 개체를 생성할 개체를 선택합니다.
4. 새 개체를 생성합니다.
5. 그런 다음 구성 요소 추가를 선택합니다. 구성 요소 이름 필드에 사용 사례에 맞는 이름을 입력합니다.
6. 구성 요소 유형 드롭다운 메뉴에서 이전 절차에서 만든 구성 요소 유형 ID를 선택합니다.
7. 구성 요소 정보, 구성 요소 이름을 입력하고 이전에 만든 하위 ComponentType을 선택합니다. Athena 데이터 커넥터를 사용하여 만든 ComponentType입니다.
8. 속성 섹션에서 구성 요소에 대한 athenaComponentExternalId에 입력합니다.



9. 구성 요소 추가를 선택하여 구성 요소 생성을 완료합니다.

이제 Athena 데이터 커넥터를 구성 요소 유형으로 사용하여 구성 요소를 성공적으로 생성하여 개체에 연결했습니다.

Athena 테이블 형식 데이터 커넥터 JSON 참조 사용

다음 예는 Athena 테이블 형식 데이터 커넥터에 대한 전체 JSON 참조입니다. 이를 리소스로 사용하여 사용자 지정 데이터 커넥터 및 구성 요소 유형을 만듭니다.

```
{
    "componentTypeId": "com.amazon.athena.connector",
    "description": "Athena connector for syncing tabular data",
    "workspaceId":"AmazonOwnedTypesWorkspace",
    "propertyGroups": {
        "tabularPropertyGroup": {
            "groupType": "TABULAR",
            "propertyNames": []
        }
    },
    "propertyDefinitions": {
        "athenaDataSource": {
            "dataType": { "type": "STRING" },
            "isRequiredInEntity": true
        },
        "athenaDatabase": {
            "dataType": { "type": "STRING" },
            "isRequiredInEntity": true
        },
        "athenaTable": {
            "dataType": { "type": "STRING" },
            "isRequiredInEntity": true
        },
        "athenaWorkgroup": {
            "dataType": { "type": "STRING" },
            "isRequiredInEntity": true
        },
        "athenaExternalIdColumnName": {
            "dataType": { "type": "STRING" },
            "isRequiredInEntity": true,
            "isExternalId": false
        },
        "athenaComponentExternalId": {
            "dataType": { "type": "STRING" },
            "isStoredExternally": false,
```

```
            "isRequiredInEntity": true,
            "isExternalId": true
        }
    },
    "functions": {
        "tabularDataReaderByEntity": {
            "implementedBy": {
                "isNative": true
            }
        }
    }
}
```

Athena 데이터 커넥터 사용하기

Grafana에서 Athena 테이블을 사용하는 개체를 표시할 수 있습니다. 자세한 내용은 AWS IoT TwinMaker Grafana 대시보드 통합을 참조하십시오.

Athena 테이블을 만들고 사용하여 데이터를 저장하는 방법에 대한 자세한 내용은 Athena 설명서를 참조하십시오.

Athena 데이터 커넥터 문제 해결

이 주제에서는 Athena 데이터 커넥터를 구성할 때 발생할 수 있는 일반적인 문제를 다룹니다.

Athena 작업 그룹 위치:

Athena 커넥터 componentType을 생성할 때는 Athena 작업그룹에 출력 위치가 설정되어 있어야 합니다. 작업 그룹 작업 방법을 참조하십시오.

IAM 역할 권한 누락:

componentType을 생성하거나, 개체에 Ca 구성 요소를 추가하거나, GetPropertyValue API를 실행할 때 AWS IoT TwinMaker; 워크스페이스 역할에 Athena API 액세스 권한이 누락되었을 수 있습니다. IAM 권한을 업데이트하려면 에 대한 서비스 역할 생성 및 관리를 AWS IoT TwinMaker 참조하세요.

Grafana에서 Athena의 테이블 형식 데이터를 시각화하십시오.

Grafana 플러그인은에 대한 API 호출 AWS IoT TwinMaker또는 Athena와의 상호 작용 없이 선택한 속성을 기반으로 정렬 및 필터링과 같은 추가 기능을 사용하여 대시보드 패널인 Grafana에서 테이블 형

식 데이터를 시각화하는 데에도 사용할 수 있습니다. 이 주제에서는 Athena 테이블 형식 데이터를 시각화하도록 Grafana를 구성하는 방법을 보여 줍니다.

사전 요구 사항

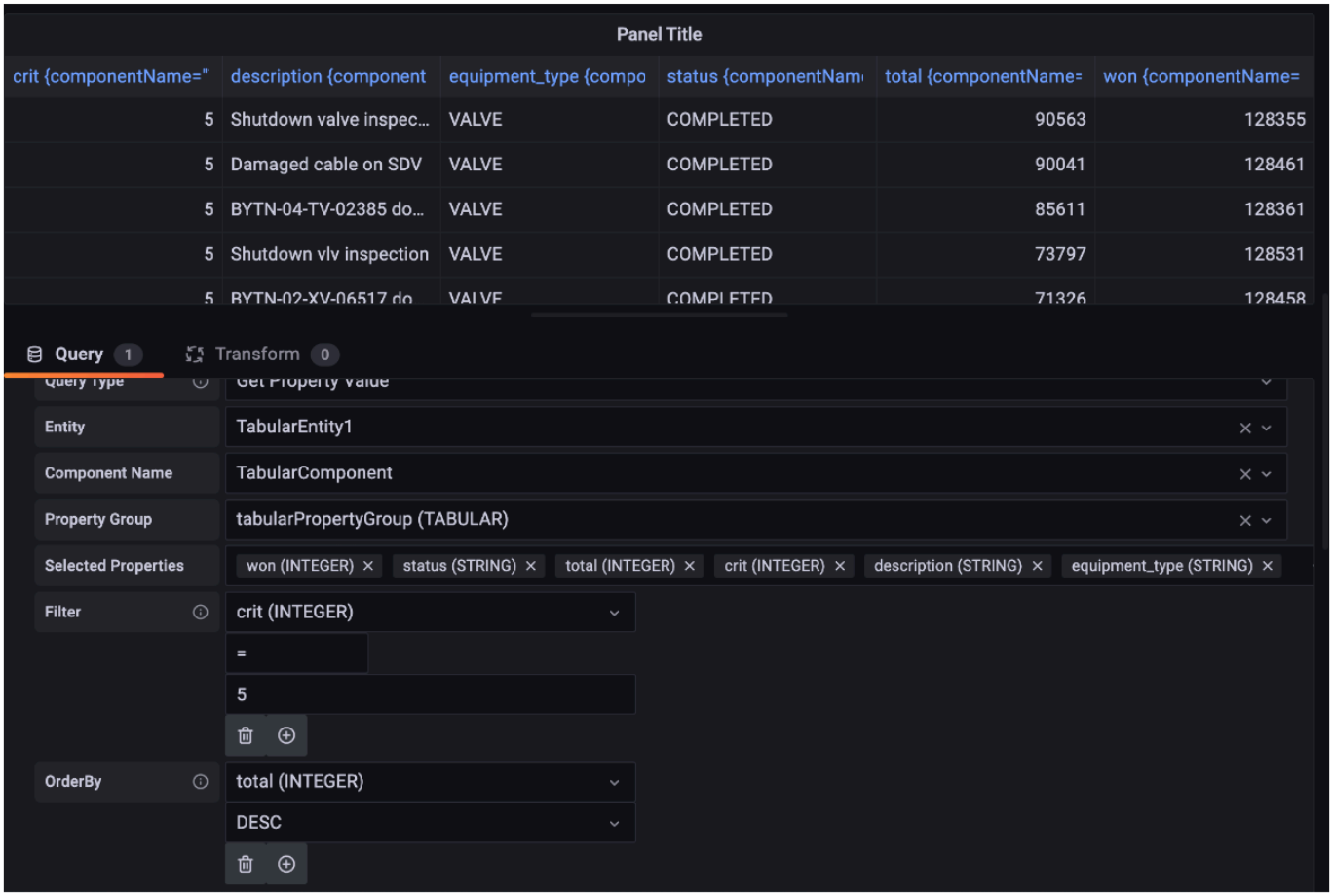
Athena 테이블 형식 데이터를 시각화하기 위해 Grafana 패널을 구성하기 전에 다음 사전 요구 사항을 검토하십시오.

- Grafana 환경을 설정했습니다. 자세한 내용은 AWS IoT TwinMaker Grafana 통합을 참조하십시오.
- Grafana 데이터 소스를 구성할 수 있습니다. 자세한 내용은 Grafana AWS IoT TwinMaker를 참조하세요.
- 새 대시보드를 만들고 새 패널을 추가하는 방법에 익숙합니다.

Grafana에서 Athena의 테이블 형식 데이터를 시각화하십시오.

이 절차는 Athena 테이블 형식 데이터를 시각화하기 위해 Grafana 패널을 설정하는 방법을 보여줍니다.

1. AWS IoT TwinMaker Grafana 대시보드를 엽니다.
2. 패널 설정에서 테이블 패널을 선택합니다.
3. 쿼리 구성에서 데이터 소스를 선택합니다.
4. 속성 값 가져오기 쿼리를 선택합니다.
5. 개체를 선택합니다.
6. Athena 기본 구성 요소 유형을 확장하는 componentType이 있는 구성 요소를 선택합니다.
7. Athena 테이블의 속성 그룹을 선택합니다.
8. 속성 그룹에서 원하는 수의 속성을 선택합니다.
9. 필터 및 속성 순서 목록을 통해 표 형식의 조건을 구성합니다. 다음 옵션이 있습니다.
    - 필터: 속성 값의 표현식을 정의하여 데이터를 필터링합니다.
    - OrderBy: 속성에 대해 데이터를 오름차순 또는 내림차순으로 반환할지 여부를 지정합니다.



AWS IoT TwinMaker 시계열 데이터 커넥터 개발

이 섹션에서는 단계별 프로세스로 시계열 데이터 커넥터를 개발하는 방법을 설명합니다. 또한 3D 모델, 개체, 구성 요소, 경보 및 커넥터를 포함하는 전체 쿠키 팩토리 샘플을 기반으로 한 시계열 데이터 커넥터 예제를 제공합니다. 쿠키 팩토리 샘플 소스는 AWS IoT TwinMaker 샘플 GitHub 저장소에서 사용할 수 있습니다.

주제

- AWS IoT TwinMaker 시계열 데이터 커넥터 사전 조건
- 시계열 데이터 커넥터 배경
- 시계열 데이터 커넥터 개발
- 데이터 커넥터을 개선합니다
- 커넥터 테스트

- 보안
- AWS IoT TwinMaker 리소스 생성
- 다음에 있는 것
- AWS IoT TwinMaker 쿠키 팩토리 예제 시계열 커넥터

AWS IoT TwinMaker 시계열 데이터 커넥터 사전 조건

시계열 데이터 커넥터를 개발하기 전에 다음 작업을 완료하는 것이 좋습니다.

- AWS IoT TwinMaker 작업 영역을 생성합니다.
- AWS IoT TwinMaker 구성 요소 유형을 생성합니다.
- AWS IoT TwinMaker 개 체를 생성합니다.
- (선택 사항) 구성 요소 유형 사용 및 만들기를 읽어 보십시오.
- (선택 사항) AWS IoT TwinMaker 데이터 커넥터 인터페이스를 읽고 AWS IoT TwinMaker 데이터 커넥터에 대한 일반적인 이해를 얻으십시오.

> **Note**
>
> 완전히 구현된 커넥터의 예는 쿠키 팩토리 예제 구현을 참조하십시오.

시계열 데이터 커넥터 배경

쿠키 믹서 세트와 물 탱크가 있는 공장에서 일하고 있다고 상상해 보십시오. 다양한 시계열 지표를 확인하여 운영 상태를 모니터링할 수 있도록 이러한 물리적 개체의 AWS IoT TwinMaker 디지털 트윈을 구축하려고 합니다.

현장 센서를 설치했고 이미 측정 데이터를 Timestream 데이터베이스로 스트리밍하고 있습니다. 오버헤드를 최소화하면서 AWS IoT TwinMaker 에서 측정 데이터를 보고 구성할 수 있기를 원합니다. 시계열 데이터 커넥터를 사용하여 이 작업을 수행할 수 있습니다. 다음 이미지는 시계열 커넥터를 사용하여 채워진 예제 원격 분석 테이블을 보여줍니다.

Rows returned (1000+)

Results are paginated. Scroll through the result pages to see more query results.

Filter

1 2 3 4 5 6 7 ... 100

TelemetryAssetId	TelemetryAssetType	measure_name	time	measure_value::varchar	measure_value::double
Mixer_22_680b5b8e-1afe-4a77-87ab-834fbe5ba01e	Mixer	Temperature	2022-04-19 00:28:00.241000000	-	99.1292877197266
Mixer_20_0568f25f-116c-429c-a974-5ceec065a6ac	Mixer	RPM	2022-04-19 00:28:00.241000000	-	59.4233207702637
Mixer_22_680b5b8e-1afe-4a77-87ab-834fbe5ba01e	Mixer	RPM	2022-04-19 00:28:00.241000000	-	59.9421195983887
Mixer_24_7ff0b75b-f0fa-43f0-bc89-b96337586d00	Mixer	Temperature	2022-04-19 00:28:00.241000000	-	99.1292877197266
Mixer_25_cf42effc-ba19-48ba-bbc3-d21d2508ce31	Mixer	RPM	2022-04-19 00:28:00.241000000	-	59.8453979492188
Mixer_20_0568f25f-116c-429c-a974-5ceec065a6ac	Mixer	Temperature	2022-04-19 00:28:00.241000000	-	99.1292877197266
Mixer_24_7ff0b75b-f0fa-43f0-bc89-b96337586d00	Mixer	RPM	2022-04-19 00:28:00.241000000	-	60.4532585144043
Mixer_15_0bb566cd-d6f3-4804-9fe1-7d2abcad82d0	Mixer	RPM	2022-04-19 00:28:00.241000000	-	58.397144317627
Mixer_2_d8e76844-e739-4845-a748-a83983279376	Mixer	RPM	2022-04-19 00:28:00.241000000	-	60.206958770752
Mixer_6_b66db3d3-c144-47b5-afb9-3a0150c53456	Mixer	RPM	2022-04-19 00:28:00.241000000	-	60.206958770752

이 스크린샷에 사용된 데이터세트와 Timestream 테이블은 AWS IoT TwinMaker 샘플 GitHub 저장소에서 사용할 수 있습니다. 또한 구현에 대한 쿠키 팩토리 예제 커넥터도 참조하십시오. 이 커넥터는 위 스크린샷에 표시된 결과를 생성합니다.

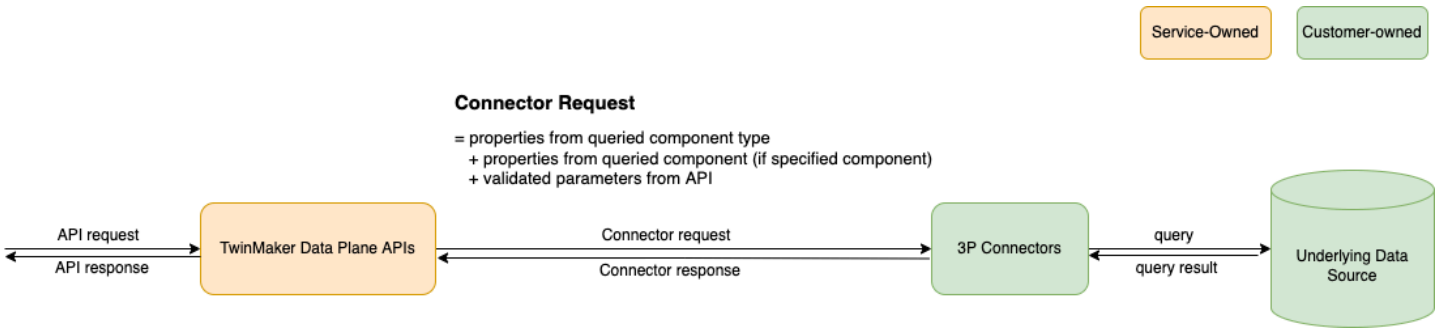
시계열 데이터 커넥터 데이터 흐름

데이터 영역 쿼리의 경우는 구성 요소 및 구성 요소 유형 정의에서 구성 요소 및 구성 요소 유형의 해당 속성을 모두 AWS IoT TwinMaker 가져옵니다.는 쿼리의 API 쿼리 파라미터와 함께 AWS Lambda 함수에 속성을 AWS IoT TwinMaker 전달합니다.

AWS IoT TwinMaker 는 Lambda 함수를 사용하여 데이터 소스의 쿼리에 액세스 및 해결하고 해당 쿼리의 결과를 반환합니다. Lambda 함수는 데이터 영역의 구성 요소 및 구성 요소 유형 속성을 사용하여 초기 요청을 해결합니다.

Lambda 쿼리 결과는 API 응답에 매핑되어 사용자에게 반환됩니다.

AWS IoT TwinMaker 는 데이터 커넥터 인터페이스를 정의하고 이를 사용하여 Lambda 함수와 상호 작용합니다. 데이터 커넥터를 사용하면 데이터 마이그레이션 작업 없이 AWS IoT TwinMaker API에서 데이터 소스를 쿼리할 수 있습니다. 다음 이미지는 이전 단락에서 설명한 기본 데이터 흐름을 간략하게 보여줍니다.



시계열 데이터 커넥터 개발

다음 절차는 기능적 시계열 데이터 커넥터로 점진적으로 구축되는 개발 모델을 간략하게 설명합니다. 기본 단계는 다음과 같습니다.

1. 유효한 기본 구성 요소 유형을 생성

   구성 요소 유형에서는 구성 요소 간에 공유되는 공통 속성을 정의합니다. 구성 요소 유형 정의에 대한 자세한 내용은 구성 요소 유형 사용 및 생성을 참조하십시오.

   AWS IoT TwinMaker 는 개체 구성 요소 모델링 패턴을 사용하므로 각 구성 요소가 개체에 연결됩니다. 각 물리적 항목을 개체로 모델링하고 자체 구성 요소 유형으로 다양한 데이터 소스를 모델링하는 것이 좋습니다.

   다음 예제는 속성이 하나인 Timestream 템플릿 구성 요소 유형을 보여 줍니다.

```
{"componentTypeId": "com.example.timestream-telemetry",
    "workspaceId": "MyWorkspace",
    "functions": {
        "dataReader": {
            "implementedBy": {
                "lambda": {
                    "arn": "lambdaArn"
                }
            }
        }
    },
    "propertyDefinitions": {
        "telemetryType": {
            "dataType": { "type": "STRING" },
            "isExternalId": false,
            "isStoredExternally": false,
            "isTimeSeries": false,
            "isRequiredInEntity": true
        },
        "telemetryId": {
            "dataType": { "type": "STRING" },
            "isExternalId": true,
            "isStoredExternally": false,
            "isTimeSeries": false,
            "isRequiredInEntity": true
        },
```

```
        "Temperature": {
            "dataType": { "type": "DOUBLE" },
            "isExternalId": false,
            "isTimeSeries": true,
            "isStoredExternally": true,
            "isRequiredInEntity": false
        }
    }
}
```

다음은 구성 요소 유형의 핵심 요소입니다.

- `telemetryId` 속성은 해당 데이터 소스에 있는 물리적 항목의 고유 키를 식별합니다. 데이터 커넥터는이 속성을 필터 조건으로 사용하여 지정된 항목과 연결된 값만 쿼리합니다. 또한 데이터 영역 API 응답에 `telemetryId` 속성 값을 포함하면 클라이언트 측에서 ID를 가져와 필요한 경우 역방향 조회를 수행할 수 있습니다.
- `lambdaArn` 필드는 구성 요소 유형이 관여하는 Lambda 함수를 식별합니다.
- `isRequiredInEntity` 플래그는 ID 생성을 강제합니다. 이 플래그는 구성 요소가 생성될 때 항목의 ID도 인스턴스화되도록 하기 위해 필요합니다.
- `TelemetryId`는 구성 요소 유형에 외부 ID로 추가되므로 Timestream 테이블에서 항목을 식별할 수 있습니다.

2. 구성 요소 유형으로 구성 요소 생성

   생성한 구성 요소 유형을 사용하려면 구성 요소를 만든 다음 데이터를 검색하려는 개체에 해당 구성 요소를 연결해야 합니다. 다음 단계는 해당 구성 요소를 만드는 프로세스를 자세히 설명합니다.

   a. AWS IoT TwinMaker 콘솔로 이동합니다.

   b. 구성 요소 유형을 만든 것과 동일한 작업 영역을 선택하여 엽니다.

   c. 개체 페이지로 이동합니다.

   d. 새 개체를 만들거나 테이블에서 기존 개체를 선택합니다.

   e. 사용할 개체를 선택한 후 구성 요소 추가를 선택하여 구성 요소 추가 페이지를 엽니다.

   f. 구성 요소에 이름을 부여하고 유형의 경우 1에서 템플릿으로 만든 구성 요소 유형을 선택합니다. 유효한 기본 구성 요소 유형을 생성하십시오.

3. 구성 요소 유형이 Lambda 커넥터를 직접적으로 호출하도록 설정

   Lambda 커넥터는 데이터 소스에 액세스하고 입력을 기반으로 쿼리 문을 생성하여 데이터 소스에 전달해야 합니다. 다음 예제에서는 이를 수행하는 JSON 요청 템플릿을 보여줍니다.

```
{
  "workspaceId": "MyWorkspace",
  "entityId": "MyEntity",
  "componentName": "TelemetryData",
  "selectedProperties": ["Temperature"],
  "startTime": "2022-08-25T00:00:00Z",
  "endTime": "2022-08-25T00:00:05Z",
  "maxResults": 3,
  "orderByTime": "ASCENDING",
  "properties": {
      "telemetryType": {
          "definition": {
              "dataType": { "type": "STRING" },
              "isExternalId": false,
              "isFinal": false,
              "isImported": false,
              "isInherited": false,
              "isRequiredInEntity": false,
              "isStoredExternally": false,
              "isTimeSeries": false
          },
          "value": {
              "stringValue": "Mixer"
          }
      },
      "telemetryId": {
          "definition": {
              "dataType": { "type": "STRING" },
              "isExternalId": true,
              "isFinal": true,
              "isImported": false,
              "isInherited": false,
              "isRequiredInEntity": true,
              "isStoredExternally": false,
              "isTimeSeries": false
          },
          "value": {
              "stringValue": "item_A001"
          }
      },
      "Temperature": {
          "definition": {
              "dataType": { "type": "DOUBLE", },
```

```
                "isExternalId": false,
                "isFinal": false,
                "isImported": true,
                "isInherited": false,
                "isRequiredInEntity": false,
                "isStoredExternally": false,
                "isTimeSeries": true
            }
        }
    }
}
```

요청의 핵심 요소는 다음과 같습니다.

- `selectedProperties`은(는) 원하는 Timestream 측정을 위해 속성으로 채우는 목록입니다.
- `startDateTime`, `startTime`, `EndDateTime` 및 `endTime` 필드는 요청의 시간 범위를 지정합니다. 이에 따라 반환되는 측정값의 샘플 범위가 결정됩니다.
- `entityId`은(는) 데이터를 쿼리하는 개체의 이름입니다.
- `componentName`은(는) 데이터를 쿼리하는 구성 요소의 이름입니다.
- `orderByTime` 필드를 사용하여 결과가 표시되는 순서를 구성할 수 있습니다.

앞의 예제 요청에서는 지정된 항목의 지정된 기간 동안 선택한 시간 순서와 함께 선택한 속성에 대한 일련의 샘플을 가져올 것으로 예상합니다. 응답 설명은 다음과 같이 요약할 수 있습니다.

```
{
  "propertyValues": [
    {
      "entityPropertyReference": {
        "entityId": "MyEntity",
        "componentName": "TelemetryData",
        "propertyName": "Temperature"
      },
      "values": [
        {
          "time": "2022-08-25T00:00:00Z",
          "value": {
            "doubleValue": 588.168
          }
        },
        {
```

```
          "time": "2022-08-25T00:00:01Z",
          "value": {
            "doubleValue": 592.4224
          }
        },
        {
          "time": "2022-08-25T00:00:02Z",
          "value": {
            "doubleValue": 594.9383
          }
        }
      ]
    }
  ],
  "nextToken": "..."
}
```

4. 두 가지 속성을 갖도록 구성 요소 유형을 업데이트

   다음의 JSON 템플릿은 두 가지 속성이 있는 유효한 구성 요소 유형을 보여줍니다.

```
{
    "componentTypeId": "com.example.timestream-telemetry",
    "workspaceId": "MyWorkspace",
    "functions": {
        "dataReader": {
            "implementedBy": {
                "lambda": {
                    "arn": "lambdaArn"
                }
            }
        }
    },
    "propertyDefinitions": {
        "telemetryType": {
            "dataType": { "type": "STRING" },
            "isExternalId": false,
            "isStoredExternally": false,
            "isTimeSeries": false,
            "isRequiredInEntity": true
        },
        "telemetryId": {
            "dataType": { "type": "STRING" },
            "isExternalId": true,
```

```
            "isStoredExternally": false,
            "isTimeSeries": false,
            "isRequiredInEntity": true
        },
        "Temperature": {
            "dataType": { "type": "DOUBLE" },
            "isExternalId": false,
            "isTimeSeries": true,
            "isStoredExternally": true,
            "isRequiredInEntity": false
        },
        "RPM": {
            "dataType": { "type": "DOUBLE" },
            "isExternalId": false,
            "isTimeSeries": true,
            "isStoredExternally": true,
            "isRequiredInEntity": false
        }
    }
}
```

5. 두 번째 속성을 처리하도록 Lambda 커넥터를 업데이트

   AWS IoT TwinMaker 데이터 영역 API는 단일 요청에서 여러 속성 쿼리를 지원하고 목록을 제공하여 커넥터에 대한 단일 요청을 AWS IoT TwinMaker 따릅니다`selectedProperties`.

   다음의 JSON 요청은 이제 두 속성에 대한 요청을 지원하는 수정된 템플릿을 보여줍니다.

```
{
  "workspaceId": "MyWorkspace",
  "entityId": "MyEntity",
  "componentName": "TelemetryData",
  "selectedProperties": ["Temperature", "RPM"],
  "startTime": "2022-08-25T00:00:00Z",
  "endTime": "2022-08-25T00:00:05Z",
  "maxResults": 3,
  "orderByTime": "ASCENDING",
  "properties": {
      "telemetryType": {
          "definition": {
              "dataType": { "type": "STRING" },
              "isExternalId": false,
              "isFinal": false,
```

```
            "isImported": false,
            "isInherited": false,
            "isRequiredInEntity": false,
            "isStoredExternally": false,
            "isTimeSeries": false
        },
        "value": {
            "stringValue": "Mixer"
        }
    },
    "telemetryId": {
        "definition": {
            "dataType": { "type": "STRING" },
            "isExternalId": true,
            "isFinal": true,
            "isImported": false,
            "isInherited": false,
            "isRequiredInEntity": true,
            "isStoredExternally": false,
            "isTimeSeries": false
        },
        "value": {
            "stringValue": "item_A001"
        }
    },
    "Temperature": {
        "definition": {
            "dataType": { "type": "DOUBLE" },
            "isExternalId": false,
            "isFinal": false,
            "isImported": true,
            "isInherited": false,
            "isRequiredInEntity": false,
            "isStoredExternally": false,
            "isTimeSeries": true
        }
    },
    "RPM": {
        "definition": {
            "dataType": { "type": "DOUBLE" },
            "isExternalId": false,
            "isFinal": false,
            "isImported": true,
            "isInherited": false,
```

```
                "isRequiredInEntity": false,
                "isStoredExternally": false,
                "isTimeSeries": true
            }
        }
    }
}
```

마찬가지로 다음 예와 같이 해당 응답도 업데이트됩니다.

```
{
  "propertyValues": [
    {
      "entityPropertyReference": {
        "entityId": "MyEntity",
        "componentName": "TelemetryData",
        "propertyName": "Temperature"
      },
      "values": [
        {
          "time": "2022-08-25T00:00:00Z",
          "value": {
            "doubleValue": 588.168
          }
        },
        {
          "time": "2022-08-25T00:00:01Z",
          "value": {
            "doubleValue": 592.4224
          }
        },
        {
          "time": "2022-08-25T00:00:02Z",
          "value": {
            "doubleValue": 594.9383
          }
        }
      ]
    },
    {
      "entityPropertyReference": {
        "entityId": "MyEntity",
        "componentName": "TelemetryData",
```

```
        "propertyName": "RPM"
      },
      "values": [
        {
          "time": "2022-08-25T00:00:00Z",
          "value": {
            "doubleValue": 59
          }
        },
        {
          "time": "2022-08-25T00:00:01Z",
          "value": {
            "doubleValue": 60
          }
        },
        {
          "time": "2022-08-25T00:00:02Z",
          "value": {
            "doubleValue": 60
          }
        }
      ]
    }
  ],
  "nextToken": "..."
}
```

> **Note**
>
> 이 경우 페이지 매김의 경우 요청의 페이지 크기가 모든 속성에 적용됩니다. 즉, 쿼리에 속성이 5개이고 페이지 크기가 100인 경우 소스에 충분한 데이터 요소가 있으면 속성당 데이터 요소 100개, 총 500개의 데이터 요소가 표시될 것으로 예상됩니다.

시된 구현의 경우 GitHub의 Snowflake 커넥터 샘플을 참조하십시오.

데이터 커넥터을 개선합니다

예외 처리

Lambda 커넥터에서 예외가 발생해도 안전합니다. 데이터 영역 API 호출에서 AWS IoT TwinMaker 서비스는 Lambda 함수가 응답을 반환할 때까지 기다립니다. 커넥터 구현에서 예외가 발생하는 경우 `ConnectorFailure`는 예외 유형을 로 AWS IoT TwinMaker 변환하여 API 클라이언트가 커넥터 내에서 문제가 발생했음을 인식하도록 합니다.

페이지 매김 처리

이 예제에서 Timestream은 기본적으로 페이지 매김을 지원하는 데 도움이 되는 유틸리티 함수를 제공합니다. 그러나 SQL과 같은 일부 다른 쿼리 인터페이스의 경우, 효율적인 페이지 매김 알고리즘을 구현하려면 추가 작업이 필요할 수 있습니다. SQL 인터페이스에서 페이지 매김을 처리하는 Snowflake 커넥터 예제가 있습니다.

커넥터 응답 인터페이스를 AWS IoT TwinMaker 통해 새 토큰이 로 반환되면 API 클라이언트로 반환되기 전에 토큰이 암호화됩니다. 토큰이 다른 요청에 포함된 경우는 토큰을 Lambda 커넥터로 전달하기 전에 AWS IoT TwinMaker 해독합니다. 토큰에 민감한 정보를 추가하지 않는 것이 좋습니다.

커넥터 테스트

커넥터를 구성 요소 유형에 연결한 후에도 구현을 업데이트할 수 있지만 AWS IoT TwinMaker와(과) 통합하기 전에 Lambda 커넥터를 확인하는 것이 좋습니다.

Lambda 커넥터를 테스트하는 방법은 여러 가지가 있습니다. Lambda 콘솔에서 또는 AWS CDK의 로컬에서 Lambda 커넥터를 테스트할 수 있습니다.

Lambda 함수 테스트에 대한 자세한 내용은 Lambda 함수 테스트 및 AWS CDK 애플리케이션 로컬 테스트를 참조하세요.

보안

Timestream을 사용한 보안 모범 사례에 대한 설명서는 Timestream의 보안을 참조하십시오.

SQL 주입 방지의 예는 AWS IoT TwinMaker 샘플 GitHub 리포지토리의 다음 Python 스크립트를 참조하세요.

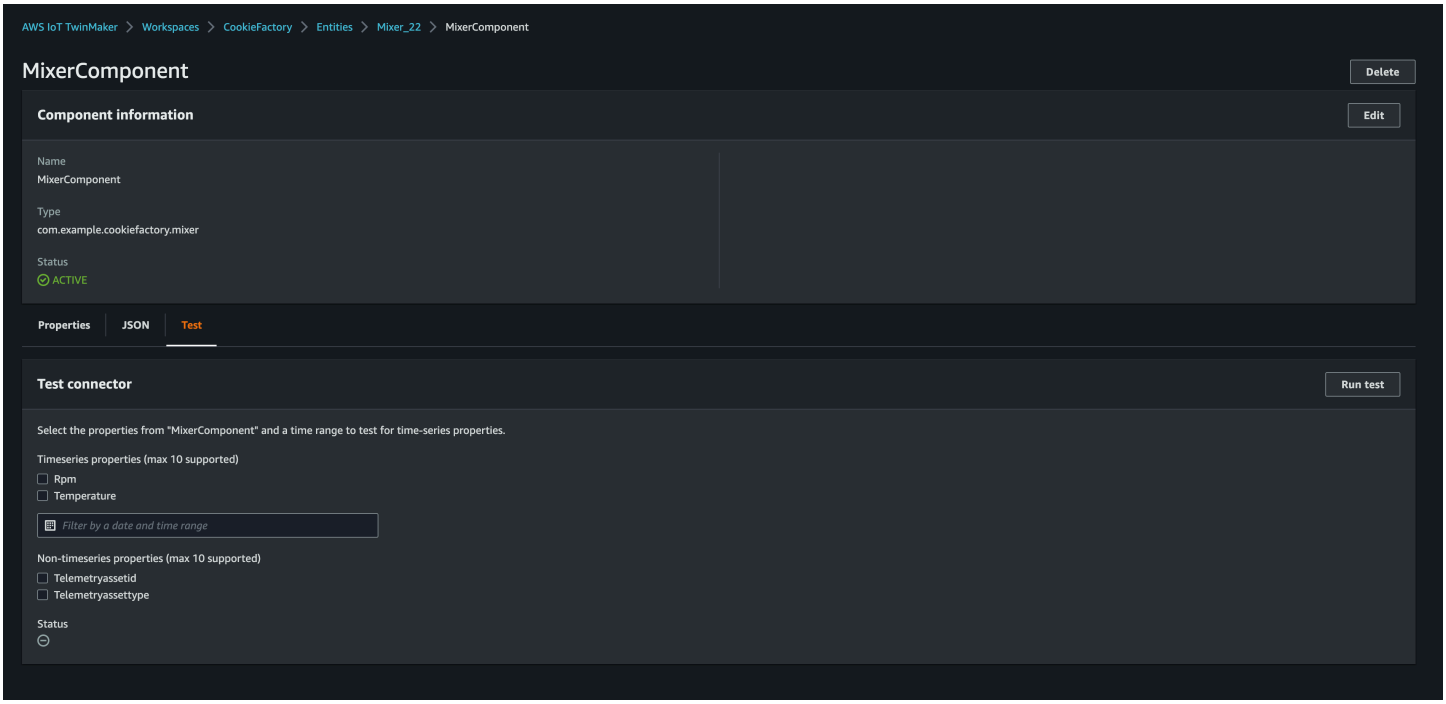
AWS IoT TwinMaker 리소스 생성

Lambda 함수를 구현한 후에는 AWS IoT TwinMaker 콘솔 또는 API를 통해 구성 요소 유형, 개체 및 구성 요소와 같은 AWS IoT TwinMaker 리소스를 생성할 수 있습니다.

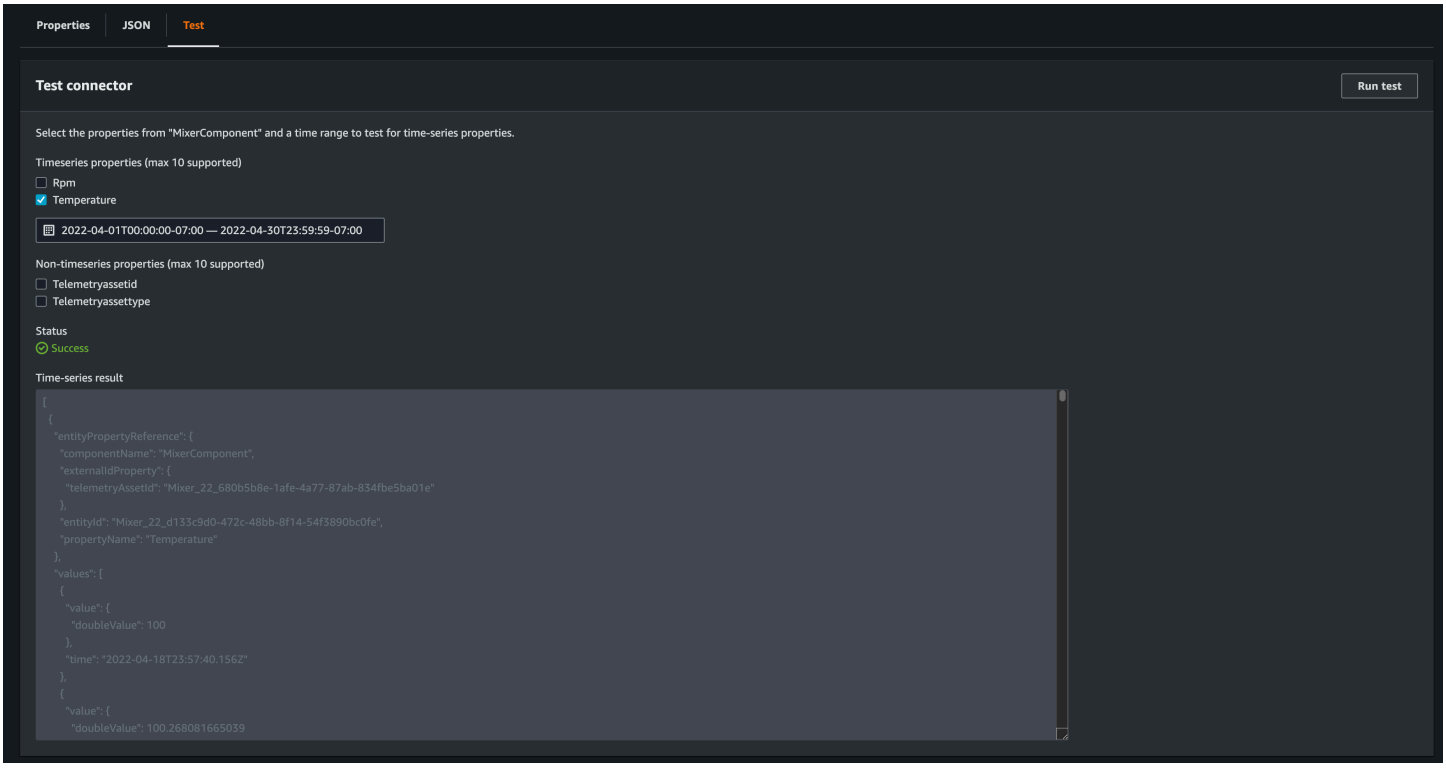
> **Note**
>
> GitHub 샘플의 설정 지침을 따르면 모든 AWS IoT TwinMaker 리소스를 자동으로 사용할 수 있습니다. AWS IoT TwinMaker GitHub 샘플에서 구성 요소 유형 정의를 확인할 수 있습니다. 일단 구성 요소에서 구성 요소 유형을 사용한 후에는 해당 구성 요소 유형의 속성 정의 및 함수를 업데이트할 수 없습니다.

통합 테스트하기

데이터 영역 쿼리가 end-to-end로 작동하는지 확인하려면와 통합된 테스트를 사용하는 AWS IoT TwinMaker 것이 좋습니다. 이 작업은 GetPropertyValueHistory API를 통해 수행하거나 AWS IoT TwinMaker 콘솔에서 쉽게 수행할 수 있습니다.



AWS IoT TwinMaker 콘솔에서 구성 요소 세부 정보로 이동한 다음 테스트 아래에 구성 요소의 모든 속성이 나열됩니다. 콘솔의 테스트 영역에서는 시계열 속성과 비시계열 속성을 테스트할 수 있습니다. 시계열 속성의 경우 GetPropertyValueHistory API를 사용하고 non-time-series 속성의 경우 GetPropertyValue API를 사용할 수도 있습니다. Lambda 커넥터가 여러 속성 쿼리를 지원하는 경우 속성을 두 개 이상 선택할 수 있습니다.



다음에 있는 것

이제 AWS IoT TwinMaker Grafana 대시보드를 설정하여 메트릭을 시각화할 수 있습니다. 또한 AWS IoT TwinMaker 샘플 GitHub 리포지토리에서 다른 데이터 커넥터 샘플을 탐색하여 그 샘플이 사용 사례에 맞는지 확인할 수도 있습니다.

AWS IoT TwinMaker 쿠키 팩토리 예제 시계열 커넥터

완전한 쿠키 팩토리 Lambda 함수의 코드는 GitHub에서 사용할 수 있습니다. 커넥터를 구성 요소 유형에 연결한 후에도 구현을 업데이트할 수 있지만 AWS IoT TwinMaker와(과) 통합하기 전에 Lambda 커넥터를 확인하는 것이 좋습니다. Lambda 콘솔에서 또는 AWS CDK에서 로컬로 Lambda 함수를 테스트할 수 있습니다. Lambda 함수 테스트에 대한 자세한 내용은 Lambda 함수 테스트 및 AWS CDK 애플리케이션 로컬 테스트를 참조하세요.

쿠키 팩토리 구성 요소 유형의 예

구성 요소 유형에서는 구성 요소 간에 공유되는 공통 속성을 정의합니다. 쿠키 팩토리 예제의 경우 동일한 유형의 물리적 구성 요소는 동일한 측정값을 공유하므로 구성 요소 유형에서 측정값 스키마를 정의할 수 있습니다. 예를 들어, 믹서 유형은 다음 예제에 정의되어 있습니다.

```
{
    "componentTypeId": "com.example.cookiefactory.mixer"
```

```
    "propertyDefinitions": {
        "RPM": {
            "dataType": { "type": "DOUBLE" },
            "isTimeSeries": true,
            "isRequiredInEntity": false,
            "isExternalId": false,
            "isStoredExternally": true
        },
        "Temperature": {
            "dataType": { "type": "DOUBLE" },
            "isTimeSeries": true,
            "isRequiredInEntity": false,
            "isExternalId": false,
            "isStoredExternally": true
        }
    }
}
```

예를 들어 물리적 구성 요소에는 Timestream 데이터베이스의 측정값, SQL 데이터베이스의 유지 관리 레코드 또는 경보 시스템의 경보 데이터가 있을 수 있습니다. 여러 구성 요소를 만들고 이를 개체와 연결하면 여러 데이터 소스가 개체에 연결되고 개체-구성 요소 그래프가 채워집니다. 이 컨텍스트에서 각 구성 요소에는 해당 데이터 소스에서 구성 요소의 고유 키를 식별하는 `telemetryId` 속성이 필요합니다. `telemetryId` 속성을 지정하면 두 가지 이점이 있습니다. 속성을 데이터 커넥터에서 필터 조건으로 사용하여 지정된 구성 요소의 값만 쿼리할 수 있으며, 데이터 영역 API 응답에 `telemetryId` 속성 값을 포함하면 클라이언트 측에서 ID를 가져와 필요한 경우 역방향 조회를 수행할 수 있습니다.

구성 요소 유형에 `TelemetryId`를 외부 ID로 추가하면 `TimeStream` 테이블의 구성 요소가 식별됩니다.

```
{
    "componentTypeId": "com.example.cookiefactory.mixer"
    "propertyDefinitions": {
        "telemetryId": {
            "dataType": { "type": "STRING" },
            "isTimeSeries": false,
            "isRequiredInEntity": true,
            "isExternalId": true,
            "isStoredExternally": false
        },
        "RPM": {
            "dataType": { "type": "DOUBLE" },
            "isTimeSeries": true,
```

```
            "isRequiredInEntity": false,
            "isExternalId": false,
            "isStoredExternally": true
        },
        "Temperature": {
            "dataType": { "type": "DOUBLE" },
            "isTimeSeries": true,
            "isRequiredInEntity": false,
            "isExternalId": false,
            "isStoredExternally": true
        }
    }
}
```

마찬가지로 다음 JSON 예와 같이 WaterTank의 구성 요소 유형이 있습니다.

```
{
  "componentTypeId": "com.example.cookiefactory.watertank",
  "propertyDefinitions": {
    "flowRate1": {
      "dataType": { "type": "DOUBLE" },
      "isTimeSeries": true,
      "isRequiredInEntity": false,
      "isExternalId": false,
      "isStoredExternally": true
    },
    "flowrate2": {
      "dataType": { "type": "DOUBLE" },
      "isTimeSeries": true,
      "isRequiredInEntity": false,
      "isExternalId": false,
      "isStoredExternally": true
    },
    "tankVolume1": {
      "dataType": { "type": "DOUBLE" },
      "isTimeSeries": true,
      "isRequiredInEntity": false,
      "isExternalId": false,
      "isStoredExternally": true
    },
    "tankVolume2": {
      "dataType": { "type": "DOUBLE" },
      "isTimeSeries": true,
```

```
      "isRequiredInEntity": false,
      "isExternalId": false,
      "isStoredExternally": true
    },
    "telemetryId": {
      "dataType": { "type": "STRING" },
      "isTimeSeries": false,
      "isRequiredInEntity": true,
      "isExternalId": true,
      "isStoredExternally": false
    }
  }
}
```

TelemetryType은(는) 개체 범위의 속성 값을 쿼리하기 위한 경우 구성 요소 유형의 선택적 속성입니다. 예제는 AWS IoT TwinMaker 샘플 GitHub 리포지토리에서 정의된 구성 요소 유형을 참조하십시오. 동일한 테이블에 경보 유형도 포함되어 있으므로 TelemetryType이(가) 정의되고 다른 하위 유형이 공유할 수있도록 TelemetryId 및 TelemetryType와(과) 같은 공통 속성을 상위 구성 요소 유형으로 추출합니다.

Lambda 예시

Lambda 커넥터는 데이터 소스에 액세스하고 입력을 기반으로 쿼리 문을 생성하여 데이터 소스에 전달해야 합니다. Lambda로 전송된 예제 요청은 다음 JSON 예제에 나와 있습니다.

```
{
    'workspaceId': 'CookieFactory',
    'selectedProperties': ['Temperature'],
    'startDateTime': 1648796400,
    'startTime': '2022-04-01T07:00:00.000Z',
    'endDateTime': 1650610799,
    'endTime': '2022-04-22T06:59:59.000Z',
    'properties': {
        'telemetryId': {
            'definition': {
                'dataType': { 'type': 'STRING' },
                'isTimeSeries': False,
                'isRequiredInEntity': True,
                'isExternalId': True,
                'isStoredExternally': False,
                'isImported': False,
                'isFinal': False,
```

```
                'isInherited': True,
            },
            'value': {
                'stringValue': 'Mixer_22_680b5b8e-1afe-4a77-87ab-834fbe5ba01e'
            }
        }
        'Temperature': {
            'definition': {
                'dataType': { 'type': 'DOUBLE' },
                'isTimeSeries': True,
                'isRequiredInEntity': False,
                'isExternalId': False,
                'isStoredExternally': True,
                'isImported': False,
                'isFinal': False,
                'isInherited': False
            }
        }
        'RPM': {
            'definition': {
                'dataType': { 'type': 'DOUBLE' },
                'isTimeSeries': True,
                'isRequiredInEntity': False,
                'isExternalId': False,
                'isStoredExternally': True,
                'isImported': False,
                'isFinal':False,
                'isInherited': False
            }
        },
    'entityId': 'Mixer_22_d133c9d0-472c-48bb-8f14-54f3890bc0fe',
    'componentName': 'MixerComponent',
    'maxResults': 100,
    'orderByTime': 'ASCENDING'
}
```

Lambda 함수의 목표는 지정된 개체에 대한 과거 측정 데이터를 쿼리하는 것입니다.는 구성 요소 속성 맵을 AWS IoT TwinMaker 제공하며 구성 요소 ID에 대해 인스턴스화된 값을 지정해야 합니다. 예를 들어 구성 요소 유형 수준 쿼리(경보 사용 사례에 공통)를 처리하고 워크스페이스에 있는 모든 구성 요소의 경보 상태를 반환하려면 속성 맵에 구성 요소 유형 속성 정의가 있습니다.

앞의 요청과 같이 가장 간단한 경우 지정된 구성 요소에 대해 지정된 기간 동안 오름차순으로 일련의 온도 샘플을 원합니다. 쿼리문은 다음과 같이 요약될 수 있습니다.

```
...
SELECT measure_name, time, measure_value::double
    FROM {database_name}.{table_name}
    WHERE time < from_iso8601_timestamp('{request.start_time}')
    AND time >= from_iso8601_timestamp('{request.end_time}')
    AND TelemetryId = '{telemetry_id}'
    AND measure_name = '{selected_property}'
    ORDER BY time {request.orderByTime}
...
```

AWS IoT TwinMaker 장면 생성 및 편집

장면은 디지털 트윈을 3차원으로 시각화한 것입니다. 이는 디지털 트윈을 편집하는 주요 방법입니다. 알람, 시계열 데이터, 색상 오버레이, 태그 및 시각적 규칙을 장면에 추가하여 디지털 트윈 시각화를 실제 사용 사례에 맞게 조정하는 방법을 알아보십시오.

이 섹션은 다음 주제를 포함합니다.

첫 번째 장면을 만들기 전

장면은 리소스를 기반으로 디지털 트윈을 표현합니다. 이러한 리소스는 3D 모델, 데이터 또는 텍스처 파일로 구성됩니다. 리소스의 크기와 복잡성, 장면의 요소(예: 조명), 컴퓨터 하드웨어는 AWS IoT TwinMaker 장면의 성능에 영향을 미칩니다. 이 항목의 정보를 사용하여 지연과 로딩 시간을 줄이고 장면의 프레임 속도를 개선할 수 있습니다.

로 가져오기 전에 리소스 최적화 AWS IoT TwinMaker

AWS IoT TwinMaker 를 사용하여 디지털 트윈과 실시간으로 상호 작용할 수 있습니다. 장면에 대한 최상의 경험을 위해서는 실시간 환경에서 사용할 수 있도록 리소스를 최적화하는 것이 좋습니다.

3D 모델은 성능에 상당한 영향을 줄 수 있습니다. 복잡한 모델 지오메트리와 메시는 성능을 저하시킬 수 있습니다. 예를 들어, 산업용 CAD 모델은 세부 수준이 높습니다. AWS IoT TwinMaker 장면에서 사용하기 전에 이러한 모델의 메시를 압축하고 다각형 수를 줄이는 것이 좋습니다. 에 대한 새 3D 모델을 생성하는 경우 세부 수준을 설정하고 모든 모델에서 이를 유지해야 AWS IoT TwinMaker합니다. 사용 사례의 시각화나 해석에 영향을 주지 않는 모델의 세부 정보를 제거하십시오.

모델을 압축하고 파일 크기를 줄이려면 DRACO 3D 데이터 압축과 같은 오픈 소스 메시 압축 도구를 사용하십시오.

최적화되지 않은 텍스처도 성능에 영향을 줄 수 있습니다. 텍스처의 투명도가 필요하지 않은 경우 PNG 형식 대신 PEG 이미지 형식을 선택하는 것이 좋습니다. Basis Universal 텍스처 압축과 같은 오픈 소스 텍스처 압축 도구를 사용하여 텍스처 파일을 압축할 수 있습니다.

의 성능 모범 사례 AWS IoT TwinMaker

에서 최상의 성능을 얻으려면 다음 제한 사항과 모범 사례를 AWS IoT TwinMaker참고하세요.

- AWS IoT TwinMaker 장면 렌더링 성능은 하드웨어에 따라 다릅니다. 성능은 컴퓨터 하드웨어 구성에 따라 달라집니다.
- AWS IoT TwinMaker안에 있는 모든 오브젝트의 총 폴리곤 수는 1백만 개 미만으로 설정하는 것이 좋습니다.
- 장면당 총 200개의 오브젝트를 사용하는 것이 좋습니다. 장면의 오브젝트 수를 200개 이상으로 늘리면 장면 프레임 속도가 감소할 수 있습니다.
- 장면 내의 모든 고유 3D 자산의 총 크기는 100MB를 초과하지 않는 것이 좋습니다. 그렇지 않으면 브라우저와 하드웨어에 따라 로딩 시간이 느려지거나 성능이 저하될 수 있습니다.
- 장면에는 기본적으로 주변 조명이 있습니다. 장면에 조명을 추가하여 특정 오브젝트에 초점을 맞추거나 그림자를 드리울 수 있습니다. 장면당 하나의 조명을 사용하는 것이 좋습니다. 필요한 경우 조명을 사용하고 장면 내에서 실제 조명을 복제하지 마십시오.

자세히 알아보기

다음 리소스를 사용하여 장면의 성능을 향상시키는 데 사용할 수 있는 최적화 기술에 대해 자세히 알아보십시오.

- 와 함께 사용할 수 있도록 OBJ 모델을 GMTF로 변환하고 압축하는 방법 AWS IoT TwinMaker
- 웹 콘텐츠용 3D 모델 최적화
- 더 나은 WebGL 성능을 위한 장면 최적화

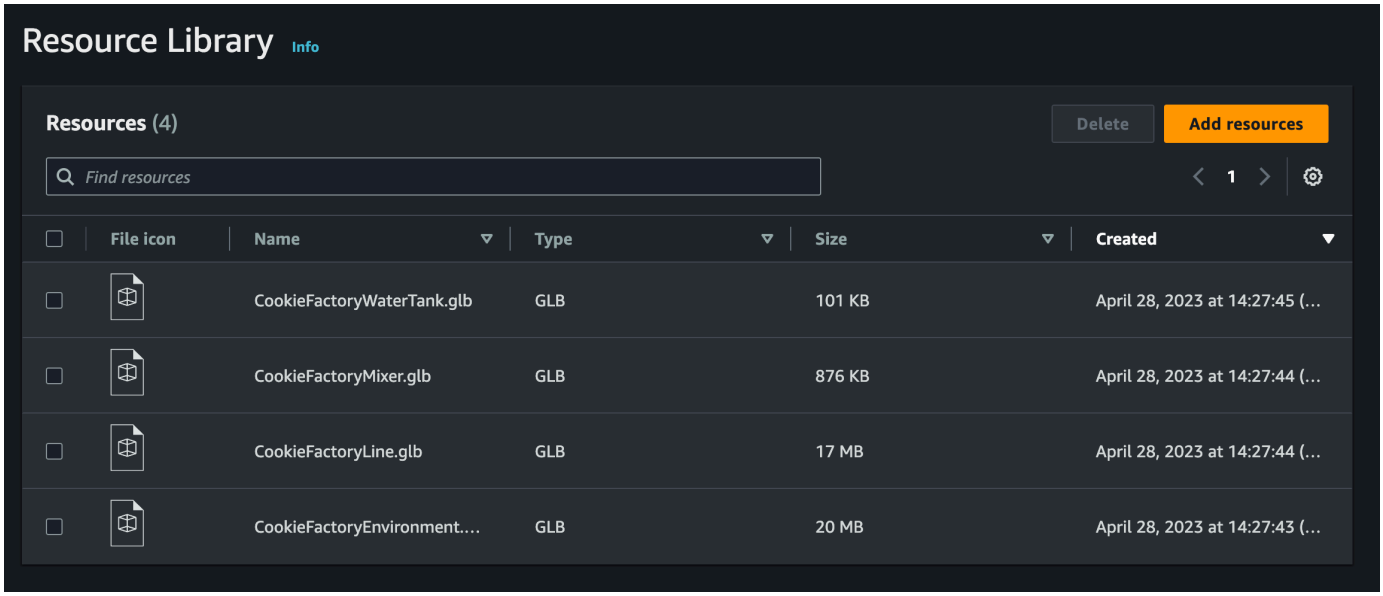
리소스 라이브러리에 AWS IoT TwinMaker 리소스 업로드

리소스 라이브러리를 사용하여 디지털 트윈 애플리케이션의 장면에 배치하려는 모든 리소스를 제어하고 관리할 수 있습니다. 리소스를 AWS IoT TwinMaker 인식하려면 Resource Library 콘솔 페이지를 사용하여 리소스를 업로드합니다.

콘솔을 사용하여 Resource Library에 파일 업로드

AWS IoT TwinMaker 콘솔을 사용하여 Resource Library에 파일을 추가하려면 다음 단계를 따르세요.

1. 왼쪽 탐색 메뉴의 Workspaces에서 Resource Library를 선택합니다.
2. 리소스 추가를 선택하고 업로드할 파일을 선택합니다.



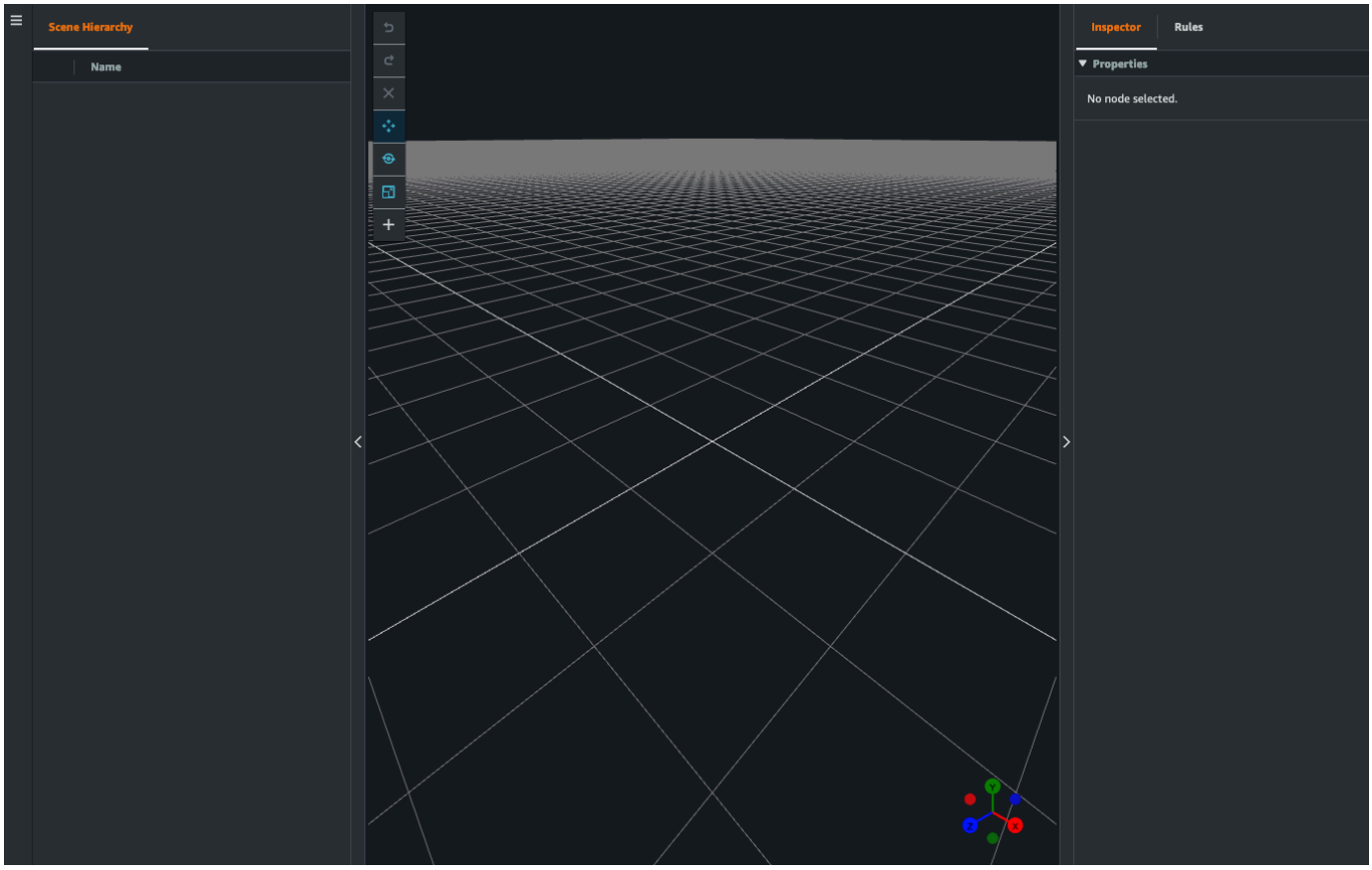
장면 생성

이 섹션에서는 디지털 트윈을 편집할 수 있는 장면을 설정해 보겠습니다. 리소스 라이브러리에 업로드된 3D 모델을 가져온 다음 위젯을 추가하고 속성 데이터를 객체에 바인딩하여 디지털 트윈을 완료할 수 있습니다. 장면 객체에는 전체 건물이나 공간 또는 물리적 위치에 배치된 개별 장비가 포함될 수 있습니다.

> **Note**
>
> 장면을 생성하기 전에 워크스페이스를 생성해야 합니다.

다음 절차에 따라 장면을 생성합니다 AWS IoT TwinMaker.

1. 장면 창을 열려면 워크스페이스의 왼쪽 탐색 창에서 장면을 선택합니다.
2. 장면 생성을 선택합니다. 새 장면 생성 창이 열립니다.
3. 새 장면 창에서 새 장면의 이름 및 설명을 입력합니다. 표준 또는 계층형 번들 요금제를 사용하는 경우 장면 유형을 선택할 수 있습니다. 동적 장면을 사용하는 것이 좋습니다.
4. 장면을 생성할 준비가 되면 장면 생성을 선택합니다. 새 장면이 열리고 작업할 준비가 되었습니다.

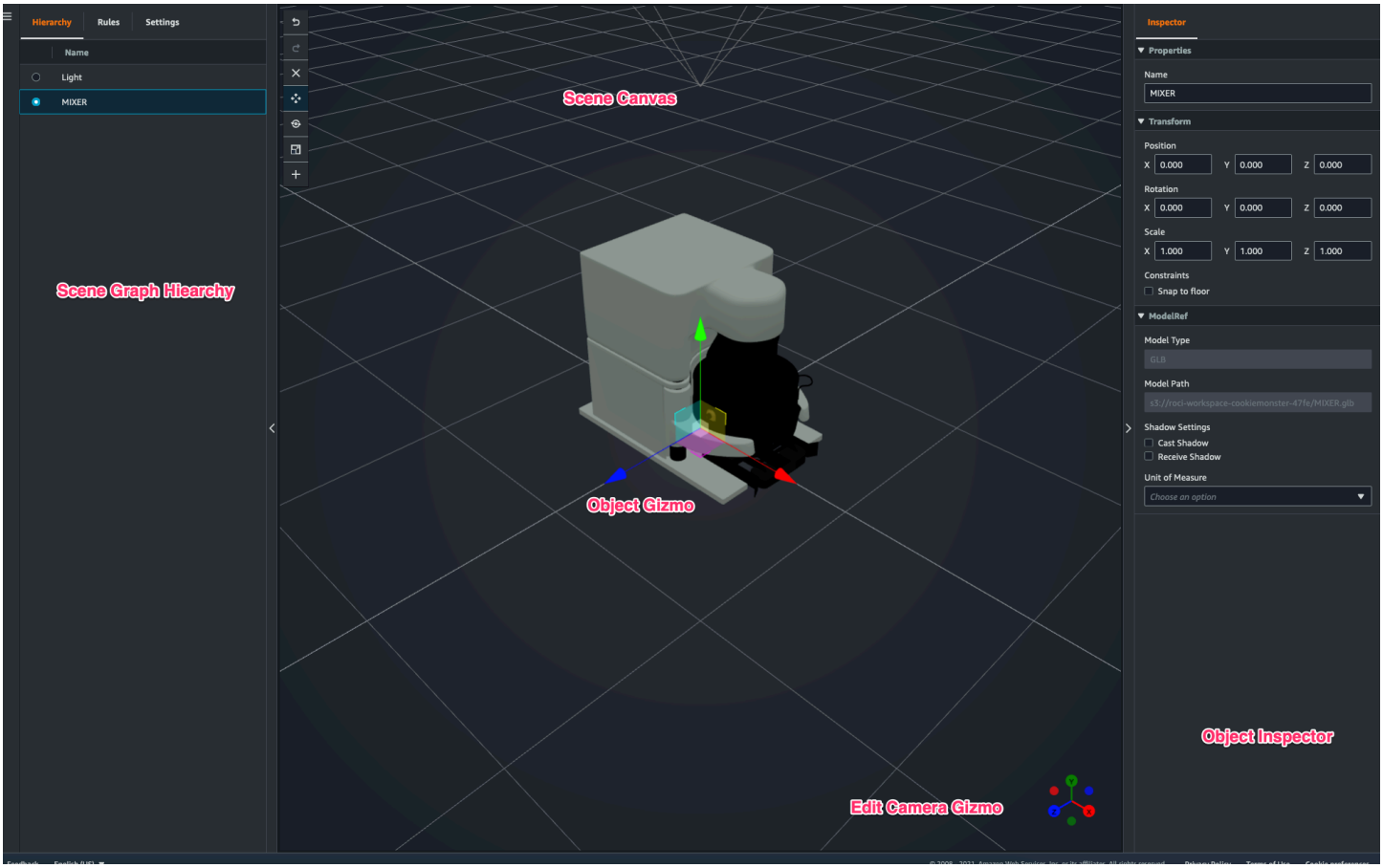


AWS IoT TwinMaker 장면에서 3D 탐색 사용

AWS IoT TwinMaker 장면에는 장면의 3D 공간을 효율적으로 탐색하는 데 사용할 수 있는 탐색 컨트롤 세트가 있습니다. 장면이 나타내는 3D 공간 및 오브젝트와 상호 작용하려면 다음 위젯 및 메뉴 옵션을 사용합니다.

- 검사기: 검사기 창을 사용하여 계층 구조에서 선택한 개체 또는 구성 요소의 속성과 설정을 보고 편집할 수 있습니다.
- 장면 캔버스: 장면 캔버스는 사용하려는 3D 리소스를 배치하고 방향을 지정할 수 있는 3D 공간입니다.

- 장면 그래프 계층: 이 패널을 사용하여 장면에 있는 모든 개체를 볼 수 있습니다. 이는 창의 왼쪽에 나타납니다.
- 오브젝트 기즈모: 이 기즈모를 사용하여 캔버스 주위로 오브젝트를 이동할 수 있습니다. 장면 캔버스에서 선택한 3D 오브젝트의 중앙에 나타납니다.
- 카메라 편집 기즈모: 카메라 편집 기즈모를 사용하여 장면 뷰 카메라의 현재 방향을 빠르게 확인하고 시야각을 수정할 수 있습니다. 장면 뷰의 오른쪽 하단 모서리에서 이 기즈모를 찾을 수 있습니다.
- 줌 컨트롤: 장면 캔버스에서 탐색하려면 마우스 오른쪽 버튼을 클릭하고 이동하려는 방향으로 드래그합니다. 회전하려면 마우스 왼쪽 버튼을 클릭하고 드래그하여 회전합니다. 확대/축소하려면 마우스의 스크롤 휠을 사용하거나 노트북의 트랙 패드에서 손가락을 모았다가 벌리십시오.



계층 창의 장면 버튼에는 다음과 같은 기능이 버튼 레이아웃 순서대로 나열되어 있습니다.

- 실행 취소: 장면에서 마지막으로 변경한 내용을 취소합니다.
- 다시 실행: 장면에서 마지막으로 변경한 내용을 다시 실행합니다.
- Plus(+): 이 버튼을 사용하면 빈 노드 추가, 3D 모델 추가, 태그 추가, 조명 추가, 모델 셰이더 추가 등의 작업에 액세스할 수 있습니다.

- 탐색 방법 변경: 장면 카메라 탐색 옵션인 궤도 및 팬에 액세스할 수 있습니다.
- 삭제: 이 버튼을 사용하면 장면에서 선택한 오브젝트를 삭제할 수 있습니다.
- 오브젝트 조작 도구: 이 버튼을 사용하면 선택한 오브젝트를 이동, 회전 및 크기 조정할 수 있습니다.

개체에 고정형 카메라 추가

AWS IoT TwinMaker 장면 내의 개체에 고정 카메라 뷰를 연결할 수 있습니다. 이 카메라는 3D 모델에 고정된 시점을 제공하므로 장면의 시점을 대상 개체로 쉽고 빠르게 전환할 수 있습니다.

1. AWS IoT TwinMaker 콘솔에서 장면으로 이동합니다.
2. 장면 계층 구조 메뉴에서 카메라를 연결할 개체를 선택합니다.
3. + 버튼을 누르고 드롭다운 옵션에서 현재 보기에서 카메라 추가를 선택합니다. 현재 시점을 사용하여 개체에 카메라를 적용합니다.
4. 검사기에서 카메라를 구성하고 다음 설정을 조정할 수 있습니다.

    - 카메라 이름
    - 카메라 위치 및 회전
    - 카메라 초점 거리
    - 확대/축소 수준
    - 근거리 및 원거리 클리핑 플레인
5. 카메라를 배치한 후 카메라에 접근할 수 있습니다. 계층 구조에서 카메라를 추가한 개체를 선택합니다. 개체 아래에 나열된 카메라 이름을 찾으십시오.
6. 개체에서 배치된 카메라를 선택하면, 장면 카메라 뷰가 배치된 카메라의 설정된 시점으로 스냅됩니다.

장면 향상 편집

AWS IoT TwinMaker 장면에는 장면에 있는 리소스의 향상된 편집 및 조작을 위한 도구 세트가 있습니다.

다음 주제에서는 AWS IoT TwinMaker 장면에서 향상된 편집 기능을 사용하는 방법을 설명합니다.

- 장면 오브젝트의 타겟 배치
- 서브모델 선택
- 장면 계층 구조에서 엔티티 편집

장면 오브젝트의 타겟 배치

AWS IoT TwinMaker 를 사용하면 장면에 객체를 정확하게 배치하고 추가할 수 있습니다. 이 향상된 편집 기능을 사용하면 장면에서 태그, 개체, 조명 및 모델을 배치할 위치를 더 잘 제어할 수 있습니다.

1. AWS IoT TwinMaker 콘솔에서 장면으로 이동합니다.
2. + 버튼을 누르고 드롭다운 옵션에서 옵션 중 하나를 선택합니다. 이는 모델, 조명, 태그 또는 + 메뉴에 있는 그 밖의 것일 수 있습니다.

   장면의 3D 공간에서 커서를 움직이면 커서 주위에 대상이 표시됩니다.
3. 대상을 사용하여 장면에 요소를 정확하게 배치합니다.

서브모델 선택

AWS IoT TwinMaker 를 사용하면 장면에서 3D 모델의 하위 모델을 선택하고 태그, 조명 또는 규칙과 같은 표준 속성을 적용할 수 있습니다.

3D 모델 파일 형식에는 모델의 하위 영역을 대형 모델 내의 하위 모델로 지정할 수 있는 메타데이터가 포함됩니다. 예를 들어 모델은 여과 시스템일 수 있으며 탱크, 파이프 또는 모터와 같은 시스템의 개별 부품은 여과 3D 모델의 하위 모델로 표시됩니다.

장면에서 지원되는 3D 파일 형식: GLB 및 GLTF.

1. AWS IoT TwinMaker 콘솔에서 장면으로 이동합니다.
2. 장면에 모델이 없는 경우 + 메뉴에서 옵션을 선택하여 모델을 추가하십시오.
3. 장면 계층 구조에 나열된 모델을 선택하면 계층 구조에 모델 아래에 하위 모델이 표시되어야 합니다.

   > **Note**
   >
   > 하위 모델이 나열되지 않는 경우 해당 모델에 하위 모델이 구성되지 않았을 가능성이 높습니다.

4. 하위 모델의 가시성을 전환하려면 계층 구조에서 하위 모델 이름 오른쪽에 있는 눈 아이콘을 누릅니다.
5. 이름이나 위치와 같은 하위 모델 데이터를 편집하려면, 하위 모델을 선택하면 장면 검사기가 열립니다. 검사기 메뉴를 사용하여 하위 모델 데이터를 업데이트하거나 변경할 수 있습니다.

6. 하위 모델에 태그, 조명, 규칙 또는 기타 속성을 추가하려면 계층 구조에서 하위 모델을 선택한 상태에서 +를 누릅니다.

장면 계층 구조에서 엔티티 편집

AWS IoT TwinMaker 장면을 사용하면 계층 구조 테이블 내에서 개체의 속성을 직접 편집할 수 있습니다. 다음 절차는 계층 메뉴를 통해 개체에 대해 수행할 수 있는 작업을 보여줍니다.

1. AWS IoT TwinMaker 콘솔에서 장면으로 이동합니다.
2. 장면 계층 구조를 열고 조작하려는 개체의 하위 요소를 선택합니다.
3. 요소를 선택한 후 + 버튼을 누르고 드롭다운에서 다음 옵션 중 하나를 선택합니다.

   - 빈 노드 추가
   - 3D 모델 추가
   - 조명 추가
   - 현재 보기에서 카메라 추가
   - 태그 추가
   - 모델 셰이더 추가
   - 모션 인디케이터 추가
4. 드롭다운에서 옵션 중 하나를 선택하면 선택 항목이 2단계에서 선택한 요소의 하위 항목으로 장면에 적용됩니다.
5. 하위 요소를 선택하고 계층 구조에서 새 상위 요소로 드래그하여 하위 요소를 재정렬하고 상위 요소를 다시 지정할 수 있습니다.

개체에 주석 추가

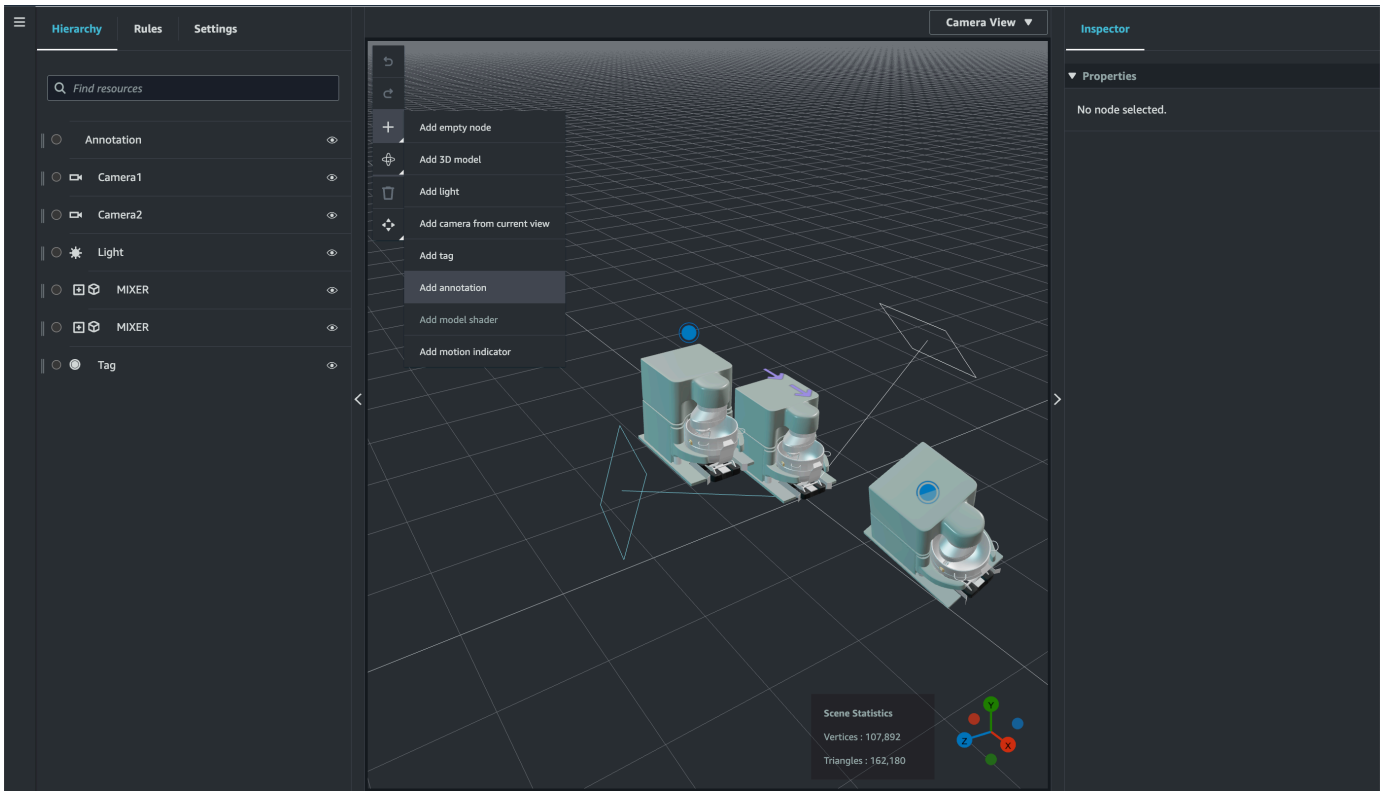
AWS IoT TwinMaker 장면 작성기를 사용하면 장면 계층 구조의 모든 요소에 주석을 달 수 있습니다. 주석은 마크다운으로 작성됩니다.

마크다운으로 작성하는 방법에 대한 자세한 내용은 마크다운 구문에 대한 공식 문서인 기본 구문을 참조하십시오.

> **Note**
>
> AWS IoT TwinMaker 주석 및 오버레이 마크다운 구문만 해당하며 HTML은 해당되지 않습니다.

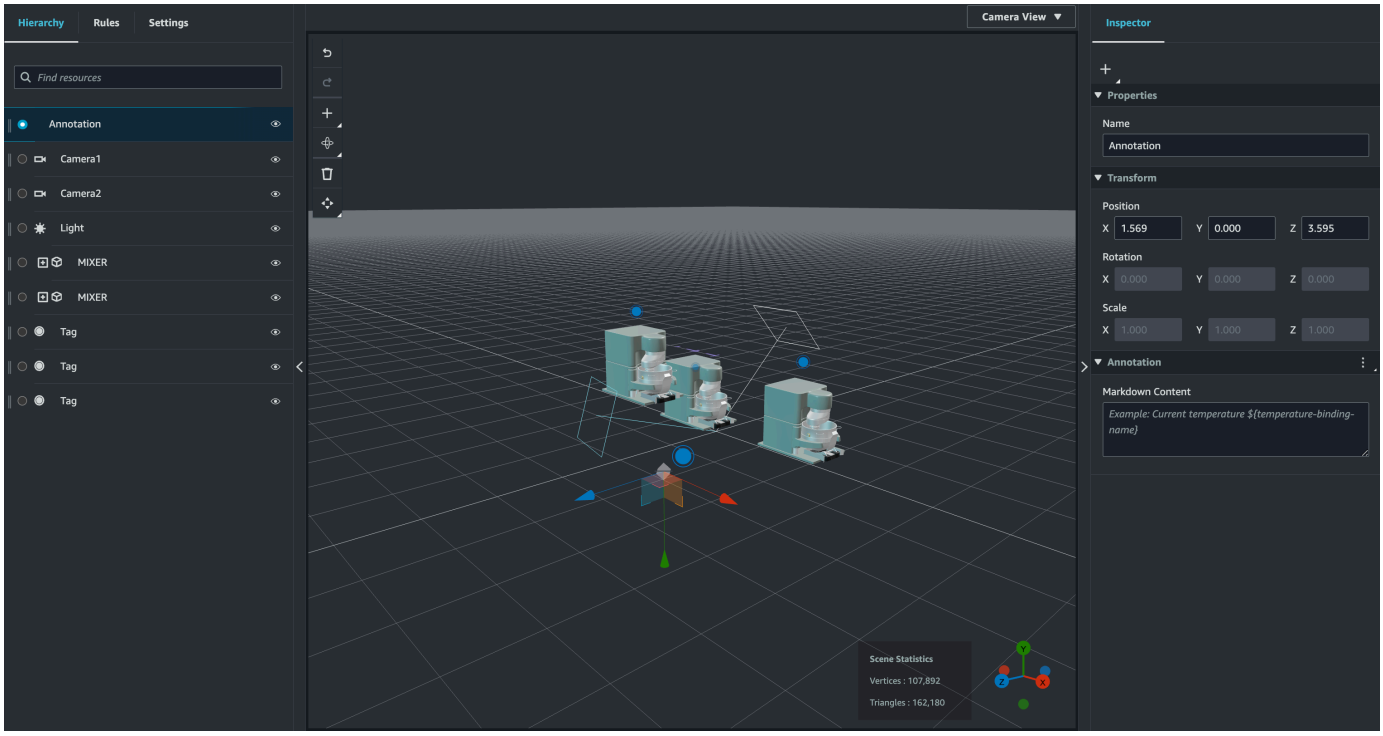
개체에 주석 추가

1. AWS IoT TwinMaker 콘솔에서 장면으로 이동합니다.
2. 장면 계층 구조에서 주석을 달고 싶은 요소를 선택합니다. 계층 구조에서 요소를 선택하지 않은 경우 루트에 주석을 추가할 수 있습니다.
3. 플러스+ 버튼을 누르고 주석 추가 옵션을 선택합니다.

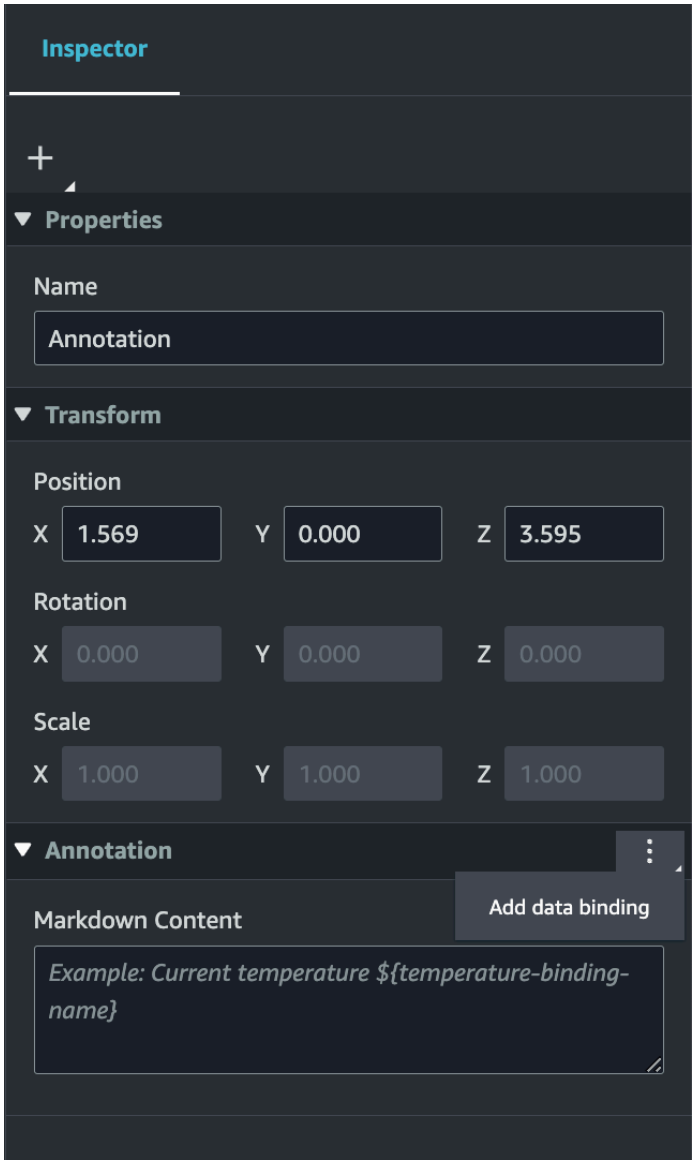


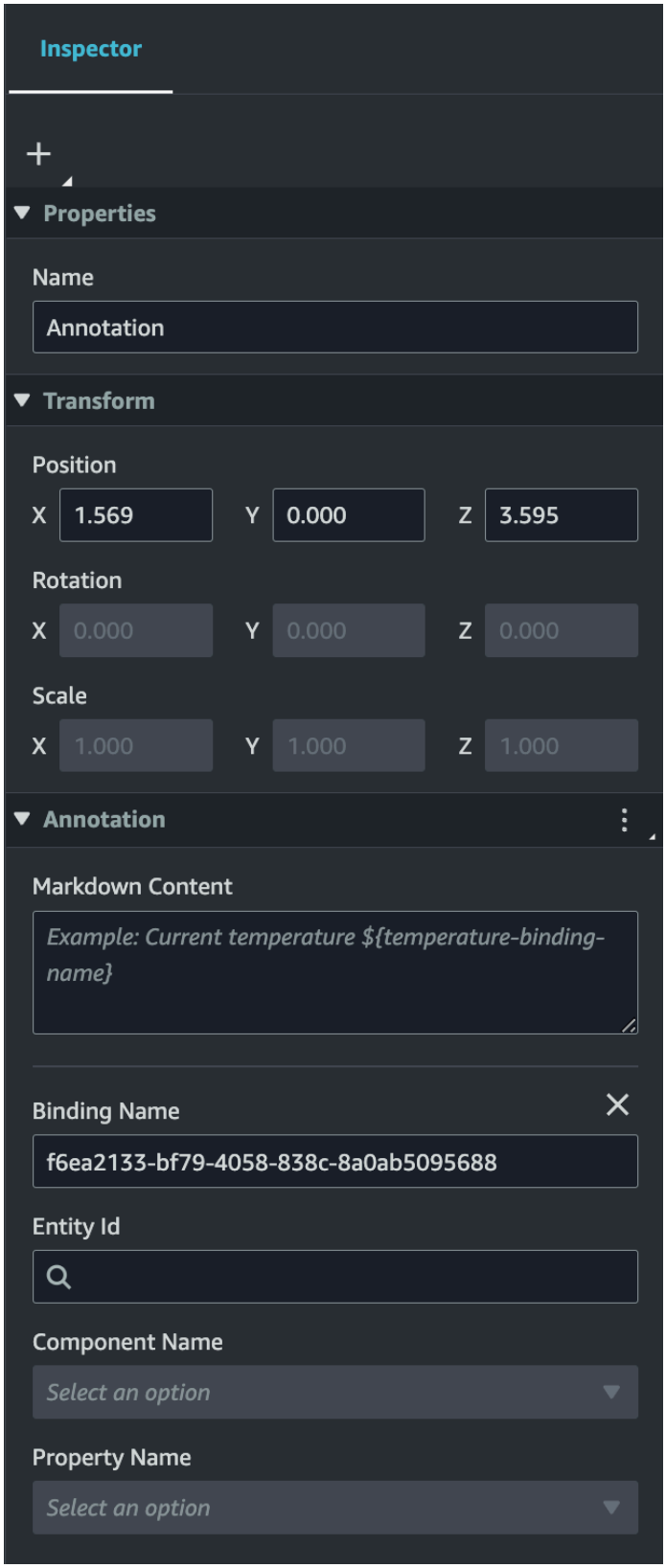
4. 왼쪽의 검사기 창에서 주석 섹션까지 아래로 스크롤합니다. 마크다운 구문을 사용하여 주석에 표시할 텍스트를 작성합니다.

   마크다운으로 작성하는 방법에 대한 자세한 내용은 마크다운 구문에 대한 공식 문서인 기본 구문을 참조하십시오.



5. AWS IoT TwinMaker 장면 데이터를 주석에 바인딩하려면 데이터 바인딩 추가를 선택하고 개체 ID를 추가한 다음 데이터를 표시할 개체의 구성 요소 이름 및 속성 이름을 선택합니다. 바인딩 이름을 업데이트하여 마크다운 변수로 사용하고 주석에 데이터를 표시할 수 있습니다.

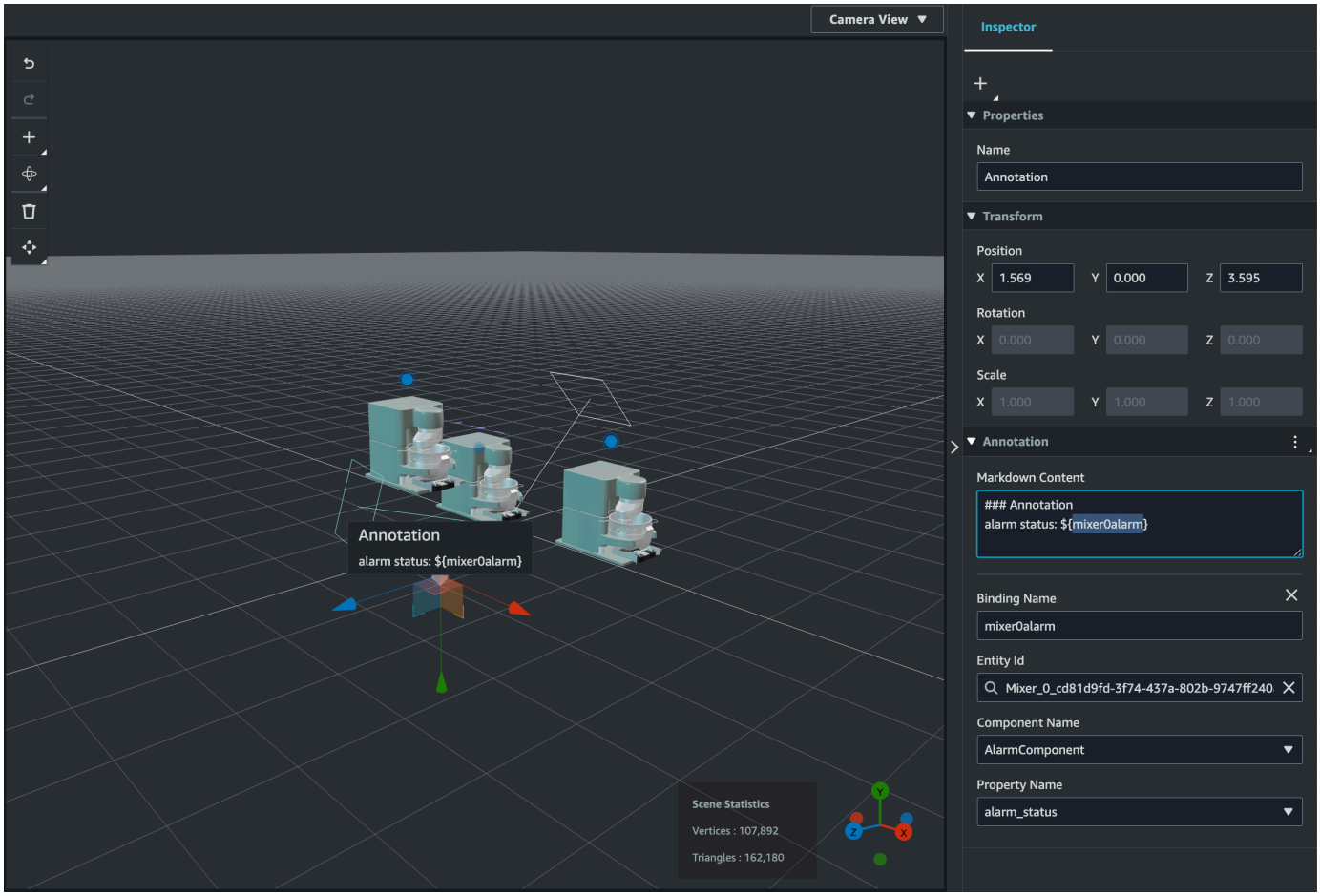

Inspector
Properties
Name
Annotation
Transform
Position
X 1.569
Y 0.000
Z 3.595
Rotation
X 0.000
Y 0.000
Z 0.000
Scale
X 1.000
Y 1.000
Z 1.000
Annotation
Add data binding
Markdown Content
Example: Current temperature ${temperature-binding-name}



6. 바인딩 이름은 주석의 변수를 나타내는 데 사용됩니다.

바인딩 이름을 입력하여 주석에서 변수 구문을 통해 엔터티 시계열의 최신 기록 값을 표시합니다 AWS IoT TwinMaker. ${*variable-name*}

예를 들어, 이 오버레이는 ${mixer0alarm} 구문이 포함된 주석에 mixer0alarm의 값을 표시합니다.



태그에 오버레이 추가

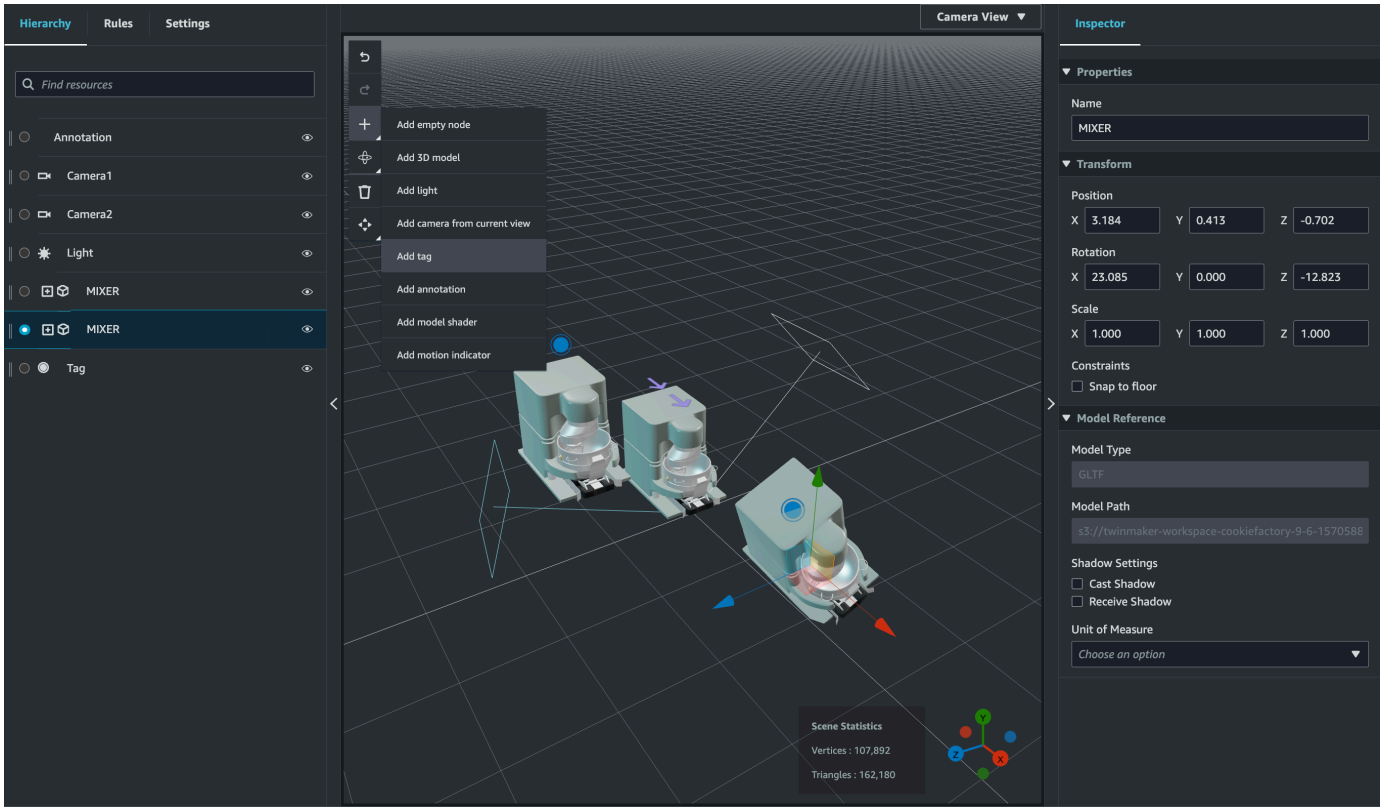
AWS IoT TwinMaker 장면에 대한 오버레이를 생성할 수 있습니다. 장면 오버레이는 태그와 연결되며 장면 개체와 연결된 중요한 데이터를 표시하는 데 사용할 수 있습니다. 오버레이는 마크다운에서 작성 및 렌더링됩니다.

마크다운으로 작성하는 방법에 대한 자세한 내용은 마크다운 구문에 대한 공식 문서인 기본 구문을 참조하십시오.

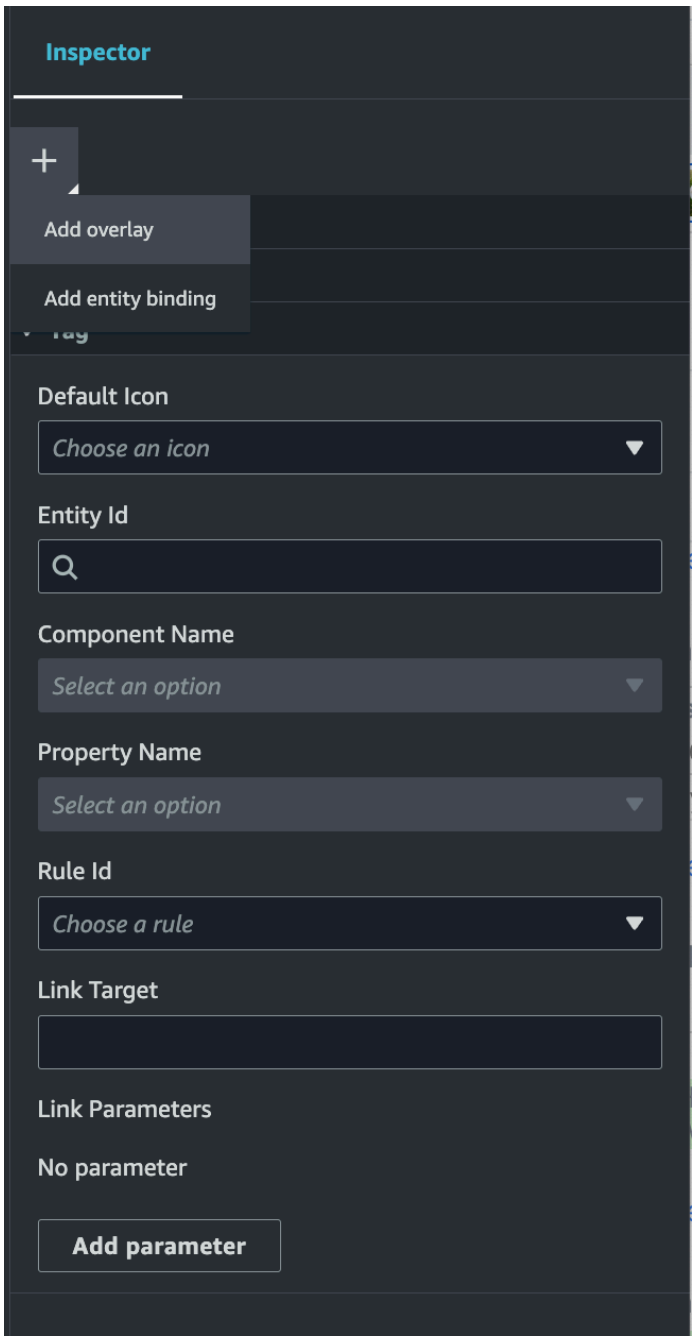
> **Note**
>
> 기본적으로 오버레이는 연결된 태그를 선택한 경우에만 장면에 표시됩니다. 장면 설정에서 이를 전환하여 모든 오버레이를 한 번에 볼 수 있습니다.

1. AWS IoT TwinMaker 콘솔에서 장면으로 이동합니다.
2. AWS IoT TwinMaker 오버레이는 태그 장면과 연결되어 있으므로 기존 태그를 업데이트하거나 새 태그를 추가할 수 있습니다.

   플러스+ 버튼을 누르고 태그 추가 옵션을 선택합니다.



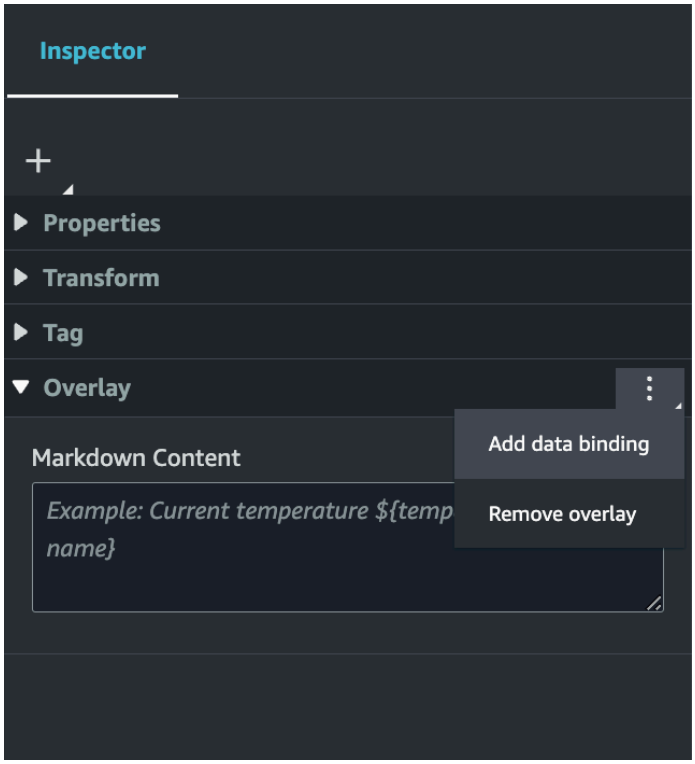
3. 오른쪽의 Inspector 패널에서 +(더하기 기호) 버튼을 선택한 다음 오버레이 추가를 선택합니다.



4. 마크다운 구문에서 오버레이에 표시할 텍스트를 작성합니다.

   마크다운으로 작성하는 방법에 대한 자세한 내용은 마크다운 구문에 대한 공식 문서인 기본 구문을 참조하십시오.

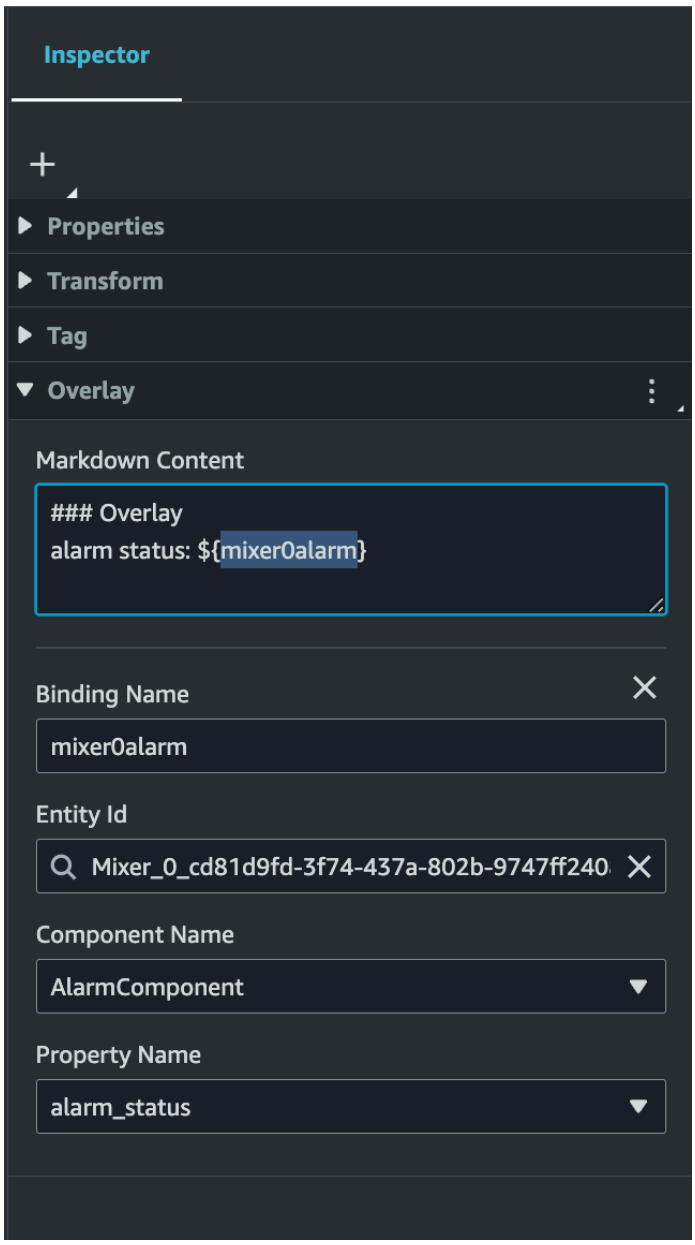
5. AWS IoT TwinMaker 장면 데이터를 오버레이에 바인딩하려면 데이터 바인딩 추가를 선택합니다.



바인딩 이름과 개체 ID를 추가한 다음 데이터를 표시할 개체의 구성 요소 이름과 속성 이름을 선택합니다.

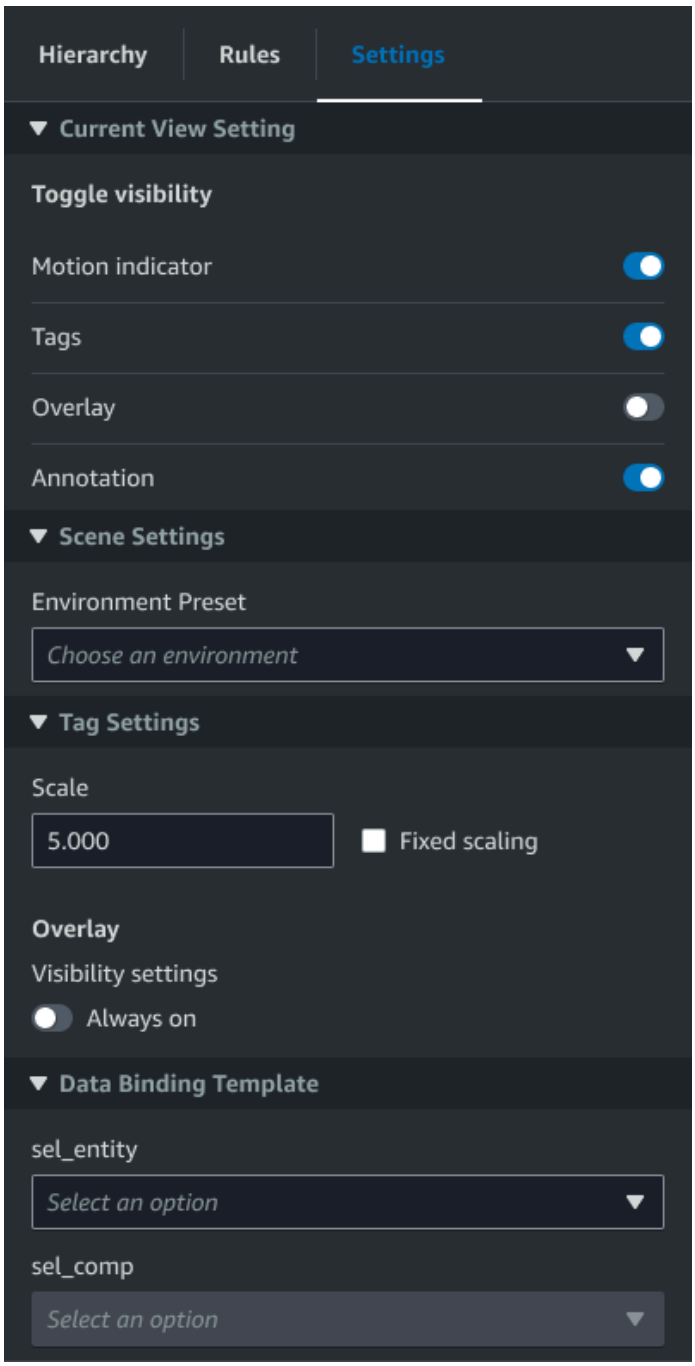
6. 의 변수 구문을 통해 오버레이에 개체 시계열 데이터의 최신 기록 값을 표시할 수 AWS IoT TwinMaker있습니다${*variable-name*}.

   예를 들어, 이 오버레이는 구문 `${mixer0alarm}`과 함께 `mixer0alarm`의 값을 오버레이에 표시합니다.



7. 오버레이 가시성을 활성화하려면 왼쪽 상단의 설정 탭을 열고 모든 오버레이가 한 번에 표시되도록 오버레이 토글이 켜져 있는지 확인합니다.

> **Note**
>
> 기본적으로 오버레이는 연결된 태그를 선택한 경우에만 장면에 표시됩니다.



장면 편집

장면을 만든 후에는 개체와 구성 요소를 장면에 추가하고 증강 위젯을 구성할 수 있습니다. 개체 구성 요소 및 위젯을 사용하여 디지털 트윈을 모델링하고 사용 사례에 맞는 기능을 제공하십시오.

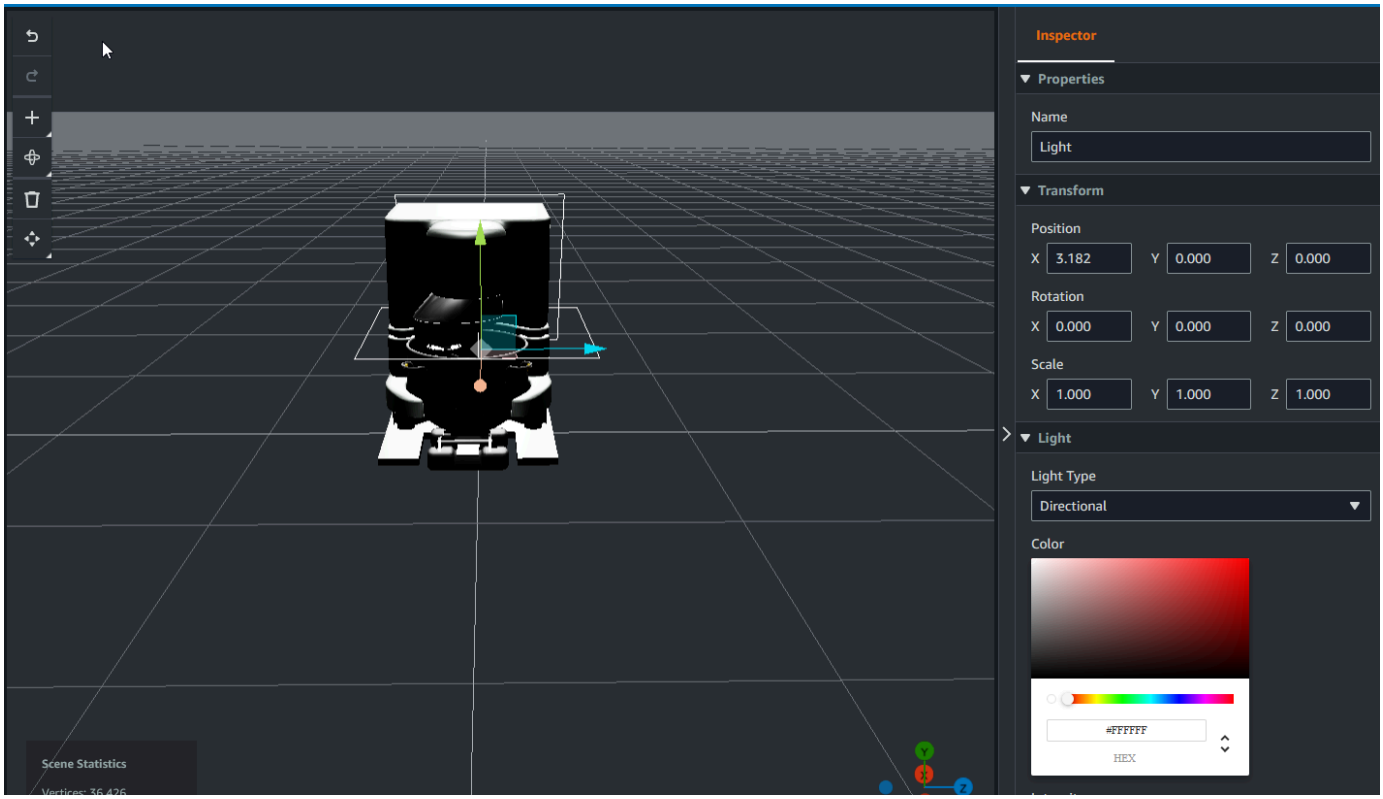
주제

장면에 모델 추가

장면에 모델을 추가하려면 다음 절차를 따르십시오.

> **Note**
>
> 장면에 모델을 추가하려면 먼저 모델을 AWS IoT TwinMaker 리소스 라이브러리에 업로드해야 합니다. 자세한 내용은 리소스 라이브러리에 AWS IoT TwinMaker 리소스 업로드을(를) 참조하십시오.

1. 장면 구성기 페이지에서 더하기(+) 기호를 선택한 다음 3D 모델 추가를 선택합니다.
2. 리소스 라이브러리에서 리소스 추가 창에서 CookieFactorMixer.glb 파일을 선택한 다음 추가를 선택합니다. 장면 구성기가 열립니다.
3. 선택 사항: 더하기(+) 기호를 선택한 다음 조명 추가를 선택합니다.
4. 각 조명 옵션을 선택하여 장면에 어떤 영향을 미치는지 확인하십시오.



> **Note**
>
> 장면에는 기본 주변 조명이 있습니다. 프레임 속도 손실을 방지하려면 장면에 배치되는 추가 조명 수를 제한하는 것이 좋습니다.

장면에 모델 셰이더 증강 UI 위젯 추가

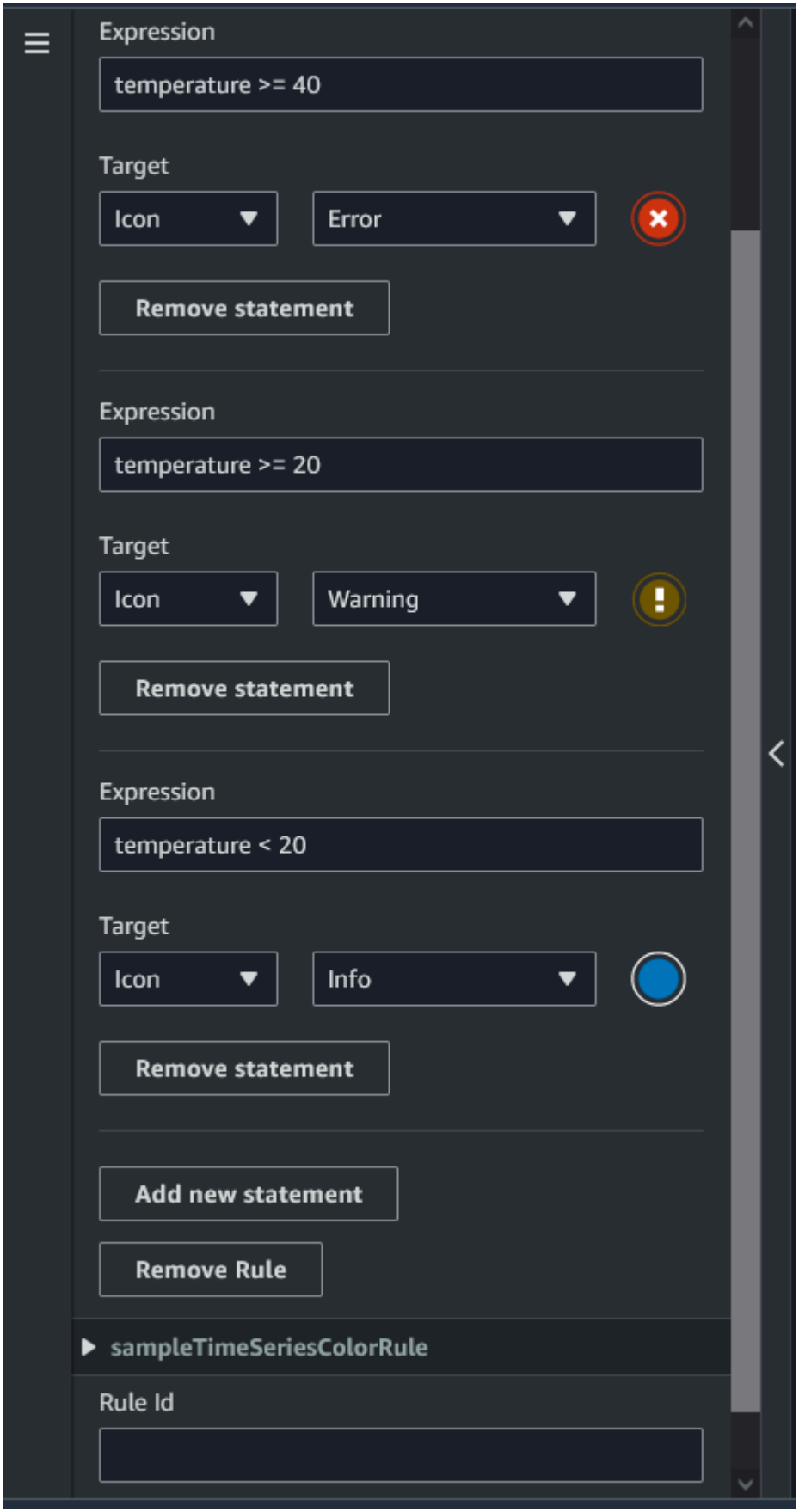
모델 셰이더 위젯은 정의한 조건에서 객체의 색상을 변경할 수 있습니다. 예를 들어 믹서의 온도 데이터를 기반으로 장면의 쿠키 믹서 색상을 변경하는 색상 위젯을 만들 수 있습니다.

다음 절차에 따라 선택한 객체에 모델 셰이더 위젯을 추가합니다.

1. 계층 구조에서 위젯을 추가할 개체를 선택합니다. + 버튼을 누른 다음 모델 셰이더를 선택합니다.
2. 새 시각적 규칙 그룹을 추가하려면 먼저 아래 지침에 따라 ColorRule을 생성한 다음 규칙 ID의 객체에 대한 Inspector 패널에서 ColorRule을 선택합니다.
3. 모델 셰이더를 바인딩할 entityID, ComponentName 및 PropertyName을 선택합니다.

장면에 대한 시각적 규칙 만들기

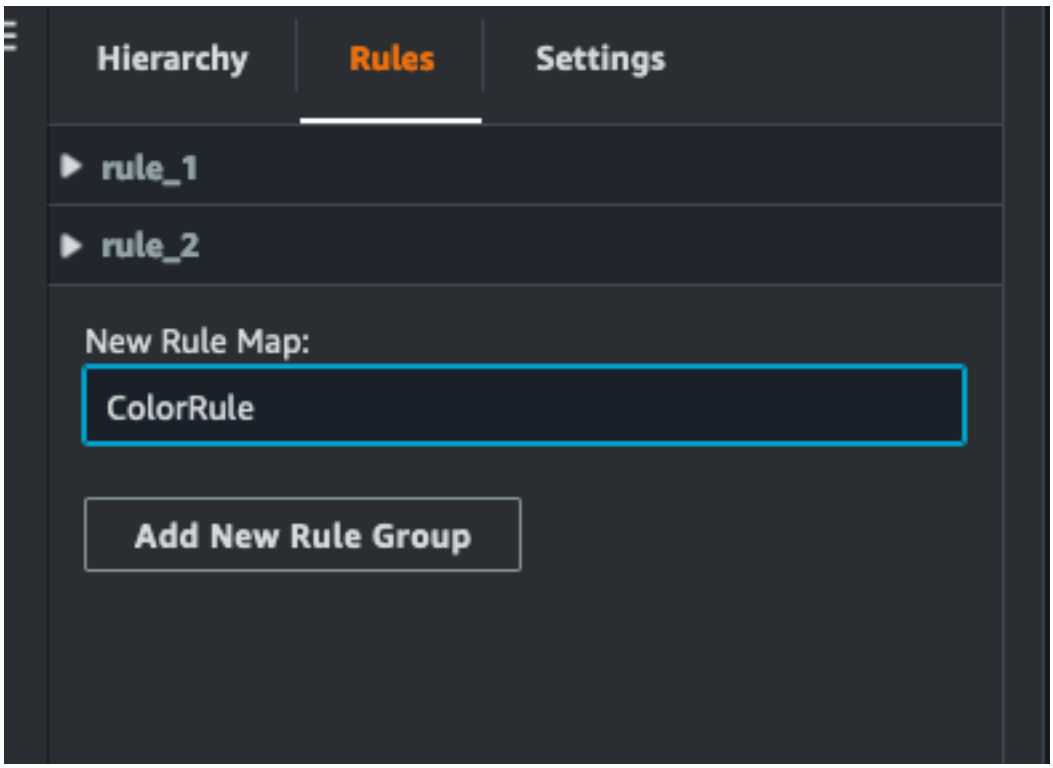
시각적 규칙 맵을 사용하여 태그 또는 모델 셰이더와 같은 증강 UI 위젯의 시각적 모양을 변경하는 데이터 기반 조건을 지정할 수 있습니다. 샘플 규칙이 제공되지만 직접 만들 수도 있습니다. 다음 예제에서는 시각적 규칙을 보여줍니다.


Expression
temperature >= 40
Target
Icon
Error
Remove statement
Expression
temperature >= 20
Target
Icon
Warning
Remove statement
Expression
temperature < 20
Target
Icon
Info
Remove statement
Add new statement
Remove Rule
sampleTimeSeriesColorRule
Rule Id

위 이미지는 ID가 '온도'인 이전에 정의된 데이터 속성을 특정 값에 대해 확인할 때의 규칙을 보여줍니다. 예를 들어 '온도'가 40보다 크거나 같으면 태그의 모양이 빨간색 원으로 변경됩니다. Grafana 대시보드에서 대상을 선택하면 동일한 데이터 소스를 사용하도록 구성된 세부 정보 패널이 채워집니다.

다음 절차는 메시 색상화 증강 UI 레이어에 새 시각적 규칙 그룹을 추가하는 방법을 보여줍니다.

1. 콘솔의 규칙 탭 아래 텍스트 필드에 ColorRule과 같은 이름을 입력하고 새 규칙 그룹 추가를 선택합니다.



2. 사용 사례에 맞는 새 규칙을 정의합니다. 예를 들어 데이터 속성 '온도'를 기반으로 하나를 생성할 수 있습니다. 여기서 보고된 값은 20 미만입니다. 규칙 표현식에 다음 구문을 사용합니다. 작음은 <, 큼은 >, 작거나 같음은 <=, 크거나 같음은 >=, 같음은 ==. (자세한 내용은 Apache Commons JEXL 구문을 참조하세요.)
3. 대상을 색상으로 설정합니다. 와 같은 색상을 정의하려면 16진수 값을 `#fcba03`사용합니다. (16진수 값에 대한 자세한 내용은 16진수를 참조하세요.)

장면에 대한 태그 만들기

태그는 장면의 특정 `x,y,z` 좌표 위치에 추가되는 주석입니다. 태그는 개체 속성을 사용하여 장면 부분을 지식 그래프에 연결합니다. 태그를 사용하여 알람과 같은 장면 내 항목의 동작이나 시각적 모양을 구성할 수 있습니다.

> **Note**
>
> 태그에 기능을 추가하려면 태그에 시각적 규칙을 적용해야 합니다.

다음 절차를 사용하여 장면에 태그를 추가합니다.

1. 계층 구조에서 오브젝트를 선택하고 + 버튼을 선택한 다음 태그 추가를 선택합니다.
2. 태그 이름을 지정합니다. 그런 다음 시각적 규칙을 적용하려면 시각적 그룹 ID를 선택합니다.
3. 드롭다운 목록에서 EntityID, ComponentName, PropertyName을 선택합니다.
4. 데이터 경로 필드를 채우려면 DataFrameLabel 생성을 선택합니다.

3D 타일 모델 형식

장면에서 3D 타일 사용

3D 장면을에 로드할 때 대기 시간이 길 AWS IoT TwinMaker 거나 복잡한 3D 모델을 탐색할 때 렌더링 성능이 좋지 않은 경우 모델을 3D 타일로 변환하는 것이 좋습니다. 이 섹션에서는 3D 타일 형식과 사용 가능한 타사 도구에 대해 설명합니다. 계속해서 3D 타일이 사용 사례에 적합한지 확인하고 시작하는 데 도움을 받으세요.

복잡한 모델 사용 사례

AWS IoT TwinMaker 장면의 3D 모델은 다음과 같은 경우 느린 로딩 시간 및 지연 탐색과 같은 성능 문제를 일으킬 수 있습니다.

- 라지: 파일 크기가 100MB보다 큽니다.
- 밀도: 수백 또는 수천 개의 개별 메시로 구성됩니다.
- 복합: 메시 지오메트리에는 복잡한 모양을 형성하는 수백만 개의 삼각형이 있습니다.

3D 타일 형식

3D 타일 형식은 모델 지오메트리를 스트리밍하고 3D 렌더링 성능을 개선하기 위한 솔루션입니다. 장면에서 3D 모델을 즉시 로드할 수 AWS IoT TwinMaker 있으며 카메라 보기에 표시되는 내용을 기반으로 모델의 청크로 로드하여 3D 상호 작용을 최적화합니다.

3D 타일 형식은 Cesium에서 생성했습니다. Cesium에는 3D 모델을 Cesium Ion이라는 3D 타일로 변환하는 관리형 서비스가 있습니다. 이는 현재 3D 타일을 생성하는 데 가장 적합한 솔루션이며 지원되는 형식의 복잡한 모델에 권장됩니다. Cesium의 요금 페이지에서 비즈니스 요구 사항에 따라 Cesium을 등록하고 적절한 구독 플랜을 선택할 수 있습니다.

AWS IoT TwinMaker 장면에 추가할 수 있는 3D 타일 모델을 준비하려면 Cesium Ion에서 설명하는 지침을 따르세요.

- Cesium Ion으로 모델 가져오기

에 Cesium 3D 타일 업로드 AWS

모델이 3D 타일로 변환되면 모델 파일을 다운로드한 다음 AWS IoT TwinMaker 워크스페이스 Amazon S3 버킷에 업로드합니다.

1. 3D 타일 모델 아카이브를 생성하고 다운로드합니다.
2. 폴더에 아카이브의 압축을 풉니다.
3. 전체 3D 타일 폴더를 워크스페이스와 연결된 Amazon S3 버킷에 업로드합니다 AWS IoT TwinMaker . (Amazon S3 사용 설명서의 객체 업로드 참조)
4. 3D 타일 모델이 성공적으로 업로드된 경우 AWS IoT TwinMaker 리소스 라이브러리에 유형이 인 Amazon S3 폴더 경로가 표시됩니다`Tiles3D`.

> **Note**
>
> AWS IoT TwinMaker Resource Library는 3D 타일 모델 직접 업로드를 지원하지 않습니다.

에서 3D 타일 사용 AWS IoT TwinMaker

AWS IoT TwinMaker 는 워크스페이스 S3 버킷에 업로드된 모든 3D 타일 모델을 인식합니다. 모델은 동일한 Amazon S3 디렉터리에서 `tileset.json` 및 모든 종속 파일(.gltf, .b3dm, .i3dm, .cmpt, .pnts)을 사용할 수 있어야 합니다. Amazon S3 디렉터리 경로가 Resource Library에 유형과 함께 표시됩니다`Tiles3D`.

장면에 3D 타일 모델을 추가하려면 다음 단계를 따르세요.

1. 장면 구성기 페이지에서 더하기(+) 기호를 선택한 다음 3D 모델 추가를 선택합니다.

2. 리소스 라이브러리에서 리소스 추가 창에서 유형이 인 3D 타일 모델의 경로를 선택한 `Tiles3D`다음 추가를 선택합니다.
3. 캔버스를 클릭하여 장면에 모델을 배치합니다.

3D 타일 차이점

3D 타일은 현재 기하학적 및 의미론적 메타데이터를 지원하지 않습니다. 즉, 원래 모델의 메시 계층 구조를 하위 모델 선택 기능에 사용할 수 없습니다. 3D 타일 모델에 위젯을 추가할 수 있지만 모델 셰이더, 분리된 3D 변환 또는 하위 모델 메시에 대한 개체 바인딩과 같은 하위 모델에 미세 조정된 기능을 사용할 수 없습니다.

장면 배경의 컨텍스트 역할을 하는 대형 자산에는 3D 타일 변환을 사용하는 것이 좋습니다. 하위 모델을 더 세분화하고 주석을 달려면 하위 모델을 별도의 glTF/glb 자산으로 추출하여 장면에 직접 추가해야 합니다. 이는 Blender와 같은 일반적인 무료 3D 도구를 사용하여 수행할 수 있습니다.

사용 사례의 예:

- 상세한 기계실과 바닥, 전기 상자 및 파이프가 있는 공장의 1GB 모델이 있습니다. 연결된 속성 데이터가 임계값을 초과하면 전기 상자와 파이프가 빨간색으로 빛나야 합니다.
- 모델에서 상자와 파이프 메시를 격리하고 Blender를 사용하여 별도의 glTF로 내보냅니다.
- 전기 및 파이프 요소를 사용하지 않고 공장을 3D 타일 모델로 변환하고 S3에 업로드합니다.
- 3D 타일 모델과 glTF 모델을 오리진(0,0,0)의 AWS IoT TwinMaker 장면에 모두 추가합니다.
- glTF의 전기 상자 및 파이프 하위 모델에 모델 셰이더 구성 요소를 추가하여 속성 규칙에 따라 메시를 빨간색으로 만듭니다.

동적 장면

AWS IoT TwinMaker 장면은 개체 구성 요소에 장면 노드와 설정을 저장하여 지식 그래프의 성능을 활용합니다. AWS IoT TwinMaker 콘솔을 사용하여 동적 장면을 생성하여 3D 장면을 보다 쉽게 관리, 빌드 및 렌더링할 수 있습니다.

주요 기능:

- 모든 3D 장면 노드 객체, 설정 및 데이터 바인딩은 지식 그래프 쿼리를 기반으로 "동적으로" 렌더링됩니다.
- Grafana 또는 사용자 지정 애플리케이션에서 읽기 전용 장면 뷰어를 사용하는 경우 30초 간격으로 장면을 업데이트할 수 있습니다.

정적 장면과 동적 장면 비교

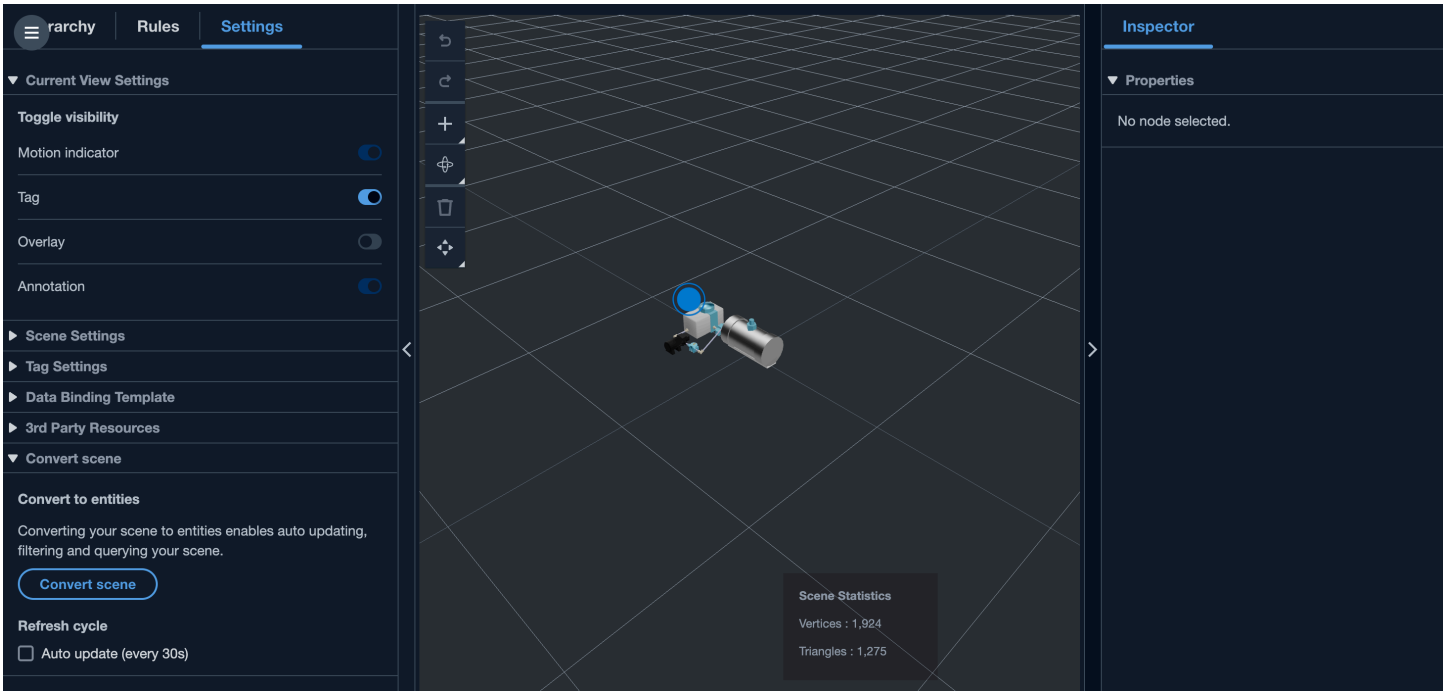
정적 장면은 모든 장면 노드 및 설정에 대한 세부 정보가 있는 S3에 저장된 장면 JSON 파일로 구성됩니다. 장면을 변경하려면 JSON 문서를 변경하고 S3에 저장해야 합니다. 기본 요금제가 있는 경우 정적 장면이 유일한 옵션입니다.

동적 장면은 장면에 대한 전역 설정이 있는 장면 JSON 파일로 구성되며, 다른 모든 장면 노드 및 노드 설정은 지식 그래프에 개체 구성 요소로 저장됩니다. 동적 장면은 표준 및 계층형 번들 요금제에서만 지원됩니다. 요금제를 업그레이드하는 방법에 대한 AWS IoT TwinMaker 가격 책정 모드 전환 자세한 내용은 섹션을 참조하세요).

다음 단계에 따라 기존 정적 장면을 동적 장면으로 변환할 수 있습니다.

- AWS IoT TwinMaker 콘솔에서 장면으로 이동합니다.
- 왼쪽 패널에서 설정 탭을 클릭합니다.
- 패널 하단의 장면 변환 섹션을 확장합니다.
- 장면 변환 버튼을 클릭한 다음 확인을 클릭합니다.

> ⚠ Warning
>
> 정적 장면에서 동적 장면으로의 변환은 되돌릴 수 없습니다.



장면 구성 요소 유형 및 개체

장면별 개체 구성 요소를 생성하기 위해 다음과 같은 1P 구성 요소 유형이 지원됩니다.

- com.amazon.iottwinmaker.3d.component.camera 카메라 위젯의 설정을 저장하는 구성 요소 유형입니다.
- com.amazon.iottwinmaker.3d.component.dataoverlay 주석 또는 태그 위젯의 오버레이에 대한 설정을 저장하는 구성 요소 유형입니다.
- com.amazon.iottwinmaker.3d.component.light 조명 위젯의 설정을 저장하는 구성 요소 유형입니다.
- com.amazon.iottwinmaker.3d.component.modelref 장면에 사용되는 3D 모델의 설정과 S3 위치를 저장하는 구성 요소 유형입니다.
- com.amazon.iottwinmaker.3d.component.modelshader 3D 모델의 모델 셰이더 설정을 저장하는 구성 요소 유형입니다.
- com.amazon.iottwinmaker.3d.component.motionindicator 모션 인디케이터 위젯의 설정을 저장하는 구성 요소 유형입니다.
- com.amazon.iottwinmaker.3d.component.submodelref 3D 모델의 하위 모델의 설정을 저장하는 구성 요소 유형입니다.
- com.amazon.iottwinmaker.3d.component.tag 태그 위젯의 설정을 저장하는 구성 요소 유형입니다.
- com.amazon.iottwinmaker.3d.node 3D 변환, 이름 및 일반 속성과 같은 장면 노드의 기본 설정을 저장하는 구성 요소 유형입니다.

동적 장면 개념

동적 장면 개체는 레이블이 지정된 글로벌 개체 아래에 저장됩니다$SCENES. 각 장면은 루트 개체와 장면 노드 계층 구조와 일치하는 하위 개체의 계층 구조로 구성됩니다. 루트 아래의 각 장면 노드에는 com.amazon.iottwinmaker.3d.node 구성 요소와 노드 유형(3D 모델, 위젯 등)에 대한 구성 요소가 있습니다.

> **Warning**
>
> 장면 개체를 수동으로 삭제하지 마십시오. 그렇지 않으면 장면이 손상된 상태일 수 있습니다. 장면을 부분적으로 또는 완전히 삭제하려면 장면 작성기 페이지를 사용하여 장면 노드를 추가 및 삭제하고, 장면 페이지를 사용하여 장면을 선택하고 삭제합니다.

AWS IoT TwinMaker UI 컴포넌트를 사용하여 사용자 지정 웹 애플리케이션 만들기

AWS IoT TwinMaker AWS IoT 애플리케이션 개발자를 위한 오픈 소스 UI 구성 요소를 제공합니다. 개발자는 이러한 UI 구성 요소를 사용하여 디지털 트윈에 사용할 수 있는 AWS IoT TwinMaker 기능을 갖춘 맞춤형 웹 애플리케이션을 구축할 수 있습니다.

AWS IoT TwinMaker UI 구성 요소는 IoT 애플리케이션 개발자가 복잡한 IoT AWS IoT 애플리케이션 개발을 단순화할 수 있도록 하는 오픈 소스 클라이언트 측 라이브러리인 Application Kit의 일부입니다.

AWS IoT TwinMaker UI 구성 요소에는 다음이 포함됩니다.

- AWS IoT TwinMaker 출처:

  데이터를 검색하고 데이터 및 디지털 트윈과 상호 작용할 수 있게 해주는 AWS IoT TwinMaker 데이터 커넥터 컴포넌트입니다.

  자세한 내용은 AWS IoT TwinMaker 소스 설명서를 참조하십시오.
- 장면 뷰어:

  `@react-three/fiber` 기반으로 구축된 3D 렌더링 구성 요소로 디지털 트윈을 렌더링하고 이를 통해 상호 작용할 수 있습니다.

  자세한 내용은 장면 뷰어 설명서를 참조하십시오.
- 동영상 플레이어:

  Kinesis Video Streams에서 비디오를 스트리밍할 수 있게 해주는 비디오 플레이어 구성 요소입니다. AWS IoT TwinMaker

  자세한 내용은 동영상 플레이어 설명서를 참조하십시오.

애플리케이션 키트 사용에 대한 자세한 내용은 AWS IoT AWS IoT 애플리케이션 키트 Github 페이지를 참조하십시오.

Application Kit를 사용하여 AWS IoT 새 웹 애플리케이션을 시작하는 방법에 대한 지침은 공식 IoT App Kit 설명서 페이지를 참조하십시오.

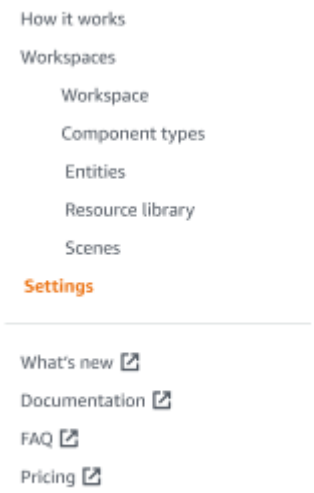
AWS IoT TwinMaker 가격 책정 모드 전환

AWS IoT TwinMaker 현재 기본, 표준 또는 계층형 번들의 세 가지 가격 책정 모드가 있습니다. 표준 가격 책정 모드가 기본 가격 책정 모드로 설정됩니다.

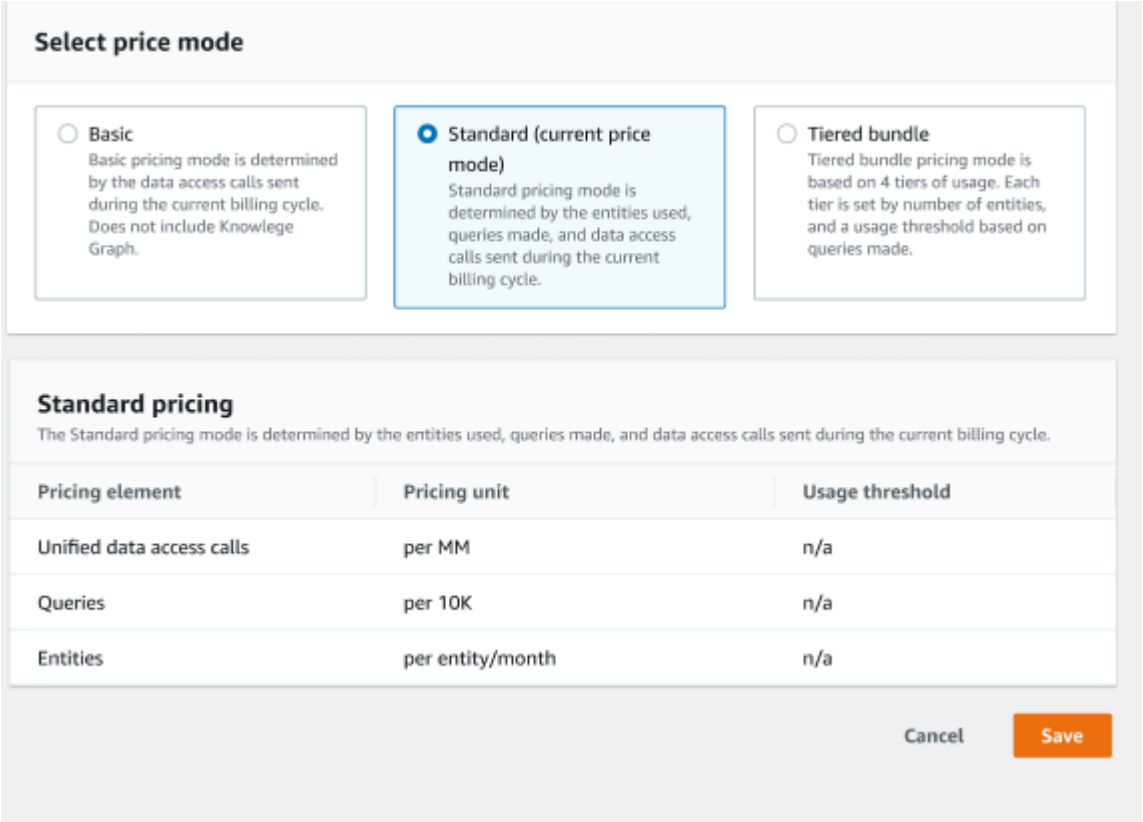
사용량 기반 가격 책정 모드에서 계층형 가격 책정 모드로 언제든지 전환할 수 있지만 변경 사항은 다음 청구 주기가 시작될 때 적용됩니다. 사용량 기반 가격 책정 모드에서 계층 기반 가격 책정 모드로 전환한 후에는 다음 세 번의 사용 주기 동안 사용량 기반 가격 책정 모드로 다시 전환할 수 없습니다. 기본에서 표준으로 전환하면 변경 사항이 즉시 적용됩니다. 세부 정보 및 비용 정보는 요금을 참조하십시오 AWS IoT TwinMaker .

이 절차는 콘솔AWS IoT TwinMaker에서 가격 책정 모드를 전환하는 방법을 보여줍니다.

1. AWS IoT TwinMaker 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 설정을 선택합니다. 가격 페이지가 열립니다.



3. 가격 변경 모드를 선택합니다.
4. 다음 스크린샷과 같이 표준 또는 티어드 번들 모드를 선택합니다.



5. 저장을 선택하여 새 가격 책정 모드를 확인합니다.
6. 이제 가격 책정 모드를 변경했습니다.

> **Note**
>
> 사용량 기반 가격 책정 모드에서 계층형 가격 책정 모드로 언제든지 전환할 수 있지만 변경 사항은 다음 청구 주기가 시작될 때 적용됩니다. 사용량 기반 가격 책정 모드에서 계층 기반 가격 책정 모드로 전환한 후에는 다음 세 번의 사용 주기 동안 사용량 기반 가격 책정 모드로 다시 전환할 수 없습니다. 기본에서 표준으로 전환하면 변경 사항이 즉시 적용됩니다.

AWS IoT TwinMaker 지식 그래프

AWS IoT TwinMaker 지식 그래프는 AWS IoT TwinMaker 워크스페이스에 포함된 모든 정보를 구성하고 시각적 그래프 형식으로 표시합니다. 개체, 구성 요소, 구성 요소 유형에 대해 쿼리를 실행하여 AWS IoT TwinMaker 리소스 간의 관계를 보여주는 시각적 그래프를 생성할 수 있습니다.

다음 주제는 지식 그래프를 사용하고 통합하는 방법을 보여줍니다.

주제

AWS IoT TwinMaker 지식 그래프 핵심 개념

이 주제는 지식 그래프 특성의 주요 개념과 어휘를 다룹니다.

지식 그래프 작동 방식:

지식 그래프는 기존 CreateEntity 또는 UpdateEntity APIs를 사용하여 개체와 해당 구성 요소 간의 관계를 생성합니다. 관계는 개체의 구성 요소에 정의된 특수 데이터 유형 RELATIONSHIP의 속성일 뿐입니다. AWS IoT TwinMaker 지식 그래프는 ExecuteQuery API를 호출하여 개체의 모든 데이터 또는 개체 간의 관계를 기반으로 쿼리를 생성합니다. 지식 그래프는 그래프 일치 구문 지원이 새로 추가된 유연한 PartiQL 쿼리 언어(많은 AWS 서비스에서 사용)를 사용하여 쿼리를 작성하는 데 도움이 됩니다. 호출이 이루어진 후 결과를 테이블로 보거나 연결된 노드 및 엣지의 그래프로 시각화할 수 있습니다.

지식 그래프 주요 용어:

- 개체 그래프: 작업 영역 내의 노드와 엣지 모음
- 노드: 작업 영역의 모든 개체가 개체 그래프에서 노드가 됩니다.
- 엣지: 개체의 구성 요소에 정의된 모든 관계 속성이 개체 그래프에서 엣지가 됩니다. 또한 개체의 parentEntityId 필드를 사용하여 정의된 계층적 상위-하위 관계도 'isChildOf' 관계 이름을 가진 개체 그래프의 엣지가 됩니다. 모든 엣지는 방향이 있는 모서리입니다.

- 관계: AWS IoT TwinMaker 관계는 개체 구성 요소의 특수한 유형의 속성입니다. CreateEntity 또는 UpdateEntity API를 사용하여 AWS IoT TwinMaker 관계를 정의하고 편집할 수 있습니다. 에서는 엔 AWS IoT TwinMaker터티의 구성 요소에 관계를 정의해야 합니다. 관계는 격리된 리소스로 정의할 수 없습니다. 관계는 한 개체에서 다른 개체로 향해야 합니다.

AWS IoT TwinMaker 지식 그래프 쿼리를 실행하는 방법

AWS IoT TwinMaker 지식 그래프를 사용하기 전에 다음 사전 조건을 완료했는지 확인합니다.

- AWS IoT TwinMaker 워크스페이스를 생성합니다. AWS IoT TwinMaker 콘솔에서 작업 영역을 생성할 수 있습니다.
- AWS IoT TwinMaker의 개체 구성 요소 시스템과 개체를 생성하는 방법을 숙지합니다. 자세한 내용은 첫 번째 개체 생성 단원을 참조하십시오.
- AWS IoT TwinMaker의 데이터 커넥터에 익숙해지세요. 자세한 내용은 AWS IoT TwinMaker 데이터 커넥터 단원을 참조하십시오.

> **Note**
>
> AWS IoT TwinMaker 지식 그래프를 사용하려면 표준 또는 계층형 번들 요금 모드에 있어야 합니다. 자세한 내용은 AWS IoT TwinMaker 가격 책정 모드 전환 단원을 참조하십시오.

다음 절차는 쿼리 작성, 실행, 저장 및 편집 방법을 보여줍니다.

쿼리 편집기 열기

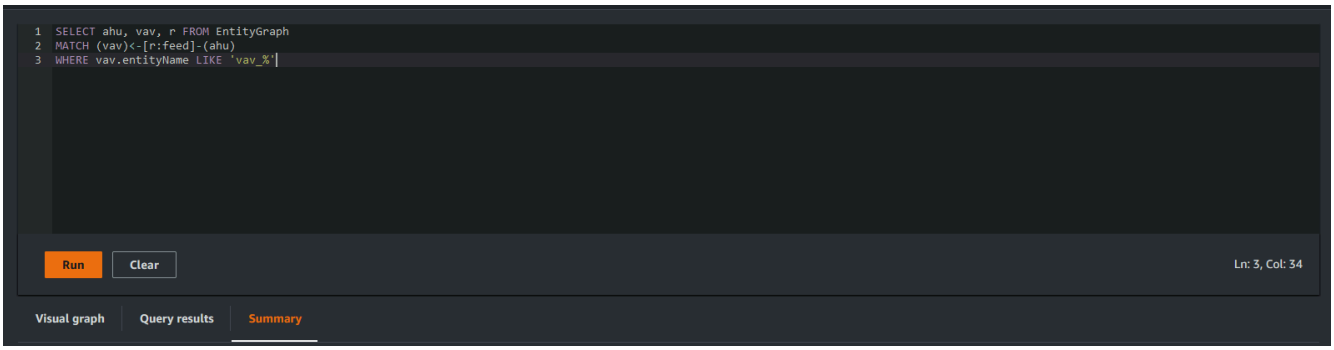
지식 그래프 쿼리 편집기로 이동하려면

1. AWS IoT TwinMaker 콘솔을 엽니다.
2. 지식 그래프를 사용하려는 작업 영역을 엽니다.
3. 왼쪽 탐색 창에서 쿼리 편집기를 선택합니다.
4. 쿼리 편집기가 열립니다. 이제 작업 영역 리소스에 대해 쿼리를 실행할 준비가 완료되었습니다.

쿼리 실행

쿼리를 실행하고 그래프를 생성하려면

1. 쿼리 편집기에서 편집기 탭을 선택하여 구문 편집기를 엽니다.
2. 편집기 공간에서 워크스페이스의 리소스에 대해 실행하려는 쿼리를 작성합니다.



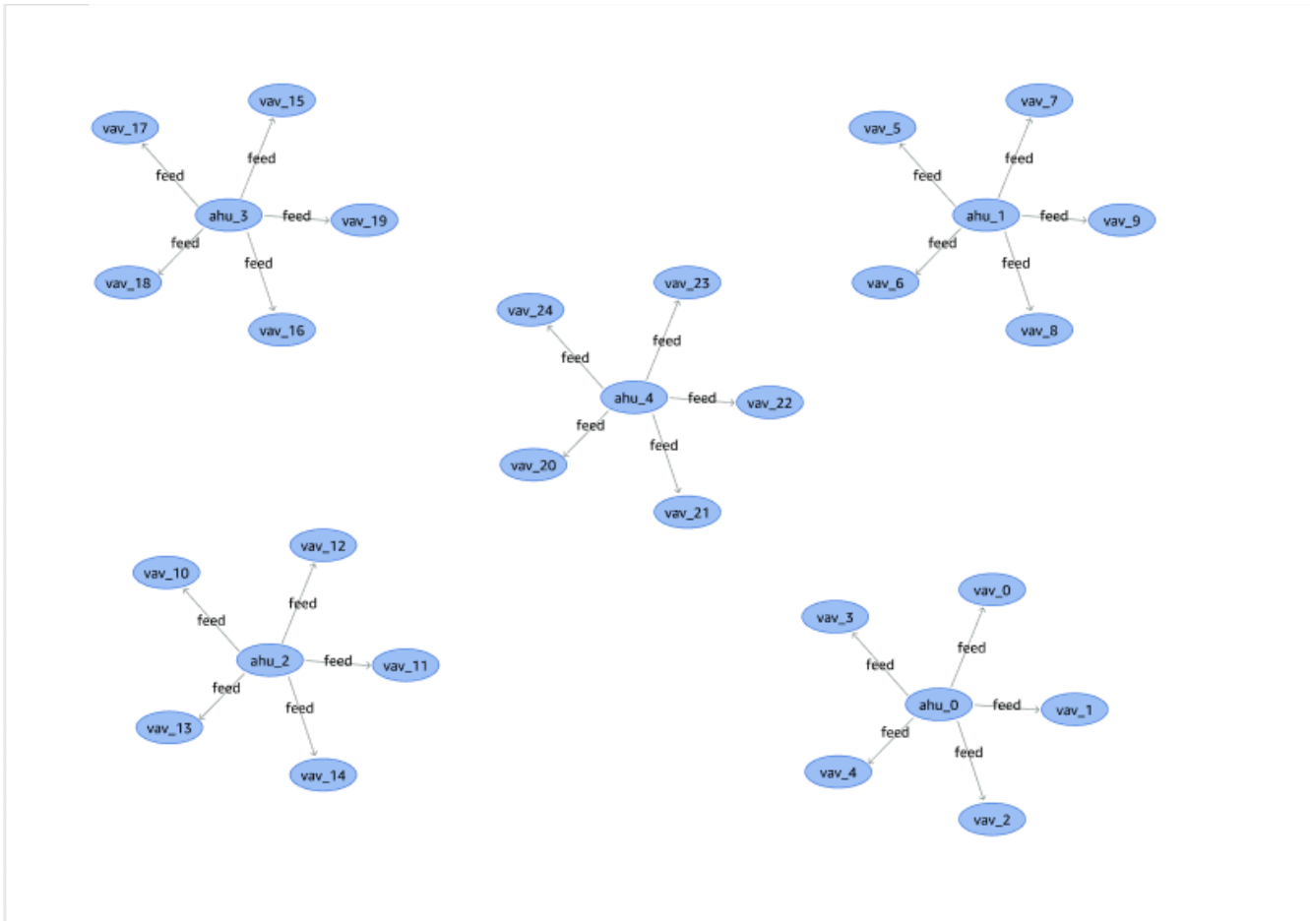
표시된 예제에서 요청은 이름vav_%에가 포함된 엔터티를 검색한 다음 다음 코드를 사용하여 해당 엔터티 간의 feed 관계를 기준으로 이러한 엔터티를 구성합니다.

```
SELECT ahu, vav, r FROM EntityGraph
MATCH (vav)<-[r:feed]-(ahu)
WHERE vav.entityName LIKE 'vav_%'
```

> **Note**
>
> 지식 그래프 구문은 PartiQL을 사용합니다. 이 구문에 대한 자세한 내용은 단원을 참조하십시오AWS IoT TwinMaker 지식 그래프 추가 리소스.

3. 쿼리 실행을 선택하여 생성한 요청을 실행합니다.

그래프는 요청에 따라 생성됩니다.



위에 표시된 예제 그래프는 2단계의 쿼리 예제를 기반으로 합니다.

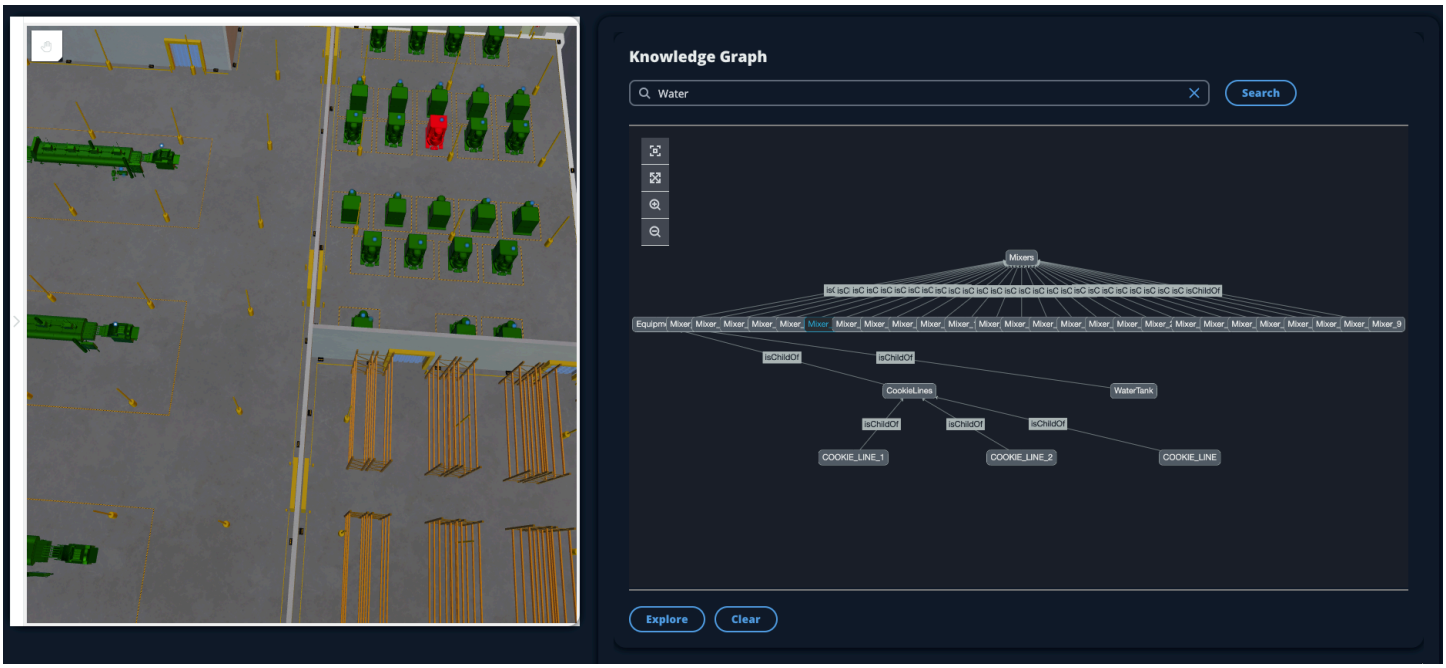
4. 쿼리의 결과도 목록에 표시됩니다. 결과를 선택하여 목록에서 쿼리 결과를 봅니다.
5. 선택적으로 다른 이름으로 내보내기를 선택하여 쿼리 결과를 JSON 또는 CSV 형식으로 내보냅니다.

여기서는 콘솔에서 지식 그래프를 사용하는 기본 방법을 다룹니다. 지식 그래프 구문을 보여주는 자세한 정보와 예는 AWS IoT TwinMaker 지식 그래프 추가 리소스을(를) 참조하십시오.

지식 그래프 장면 통합

AWS IoT 앱 키트 구성 요소를 사용하여 지식 그래프를 AWS IoT TwinMaker 장면에 통합하는 웹 애플리케이션을 구축할 수 있습니다. 이를 통해 장면 내에 있는 3D 노드(장비 또는 시스템을 나타내는 3D 모델)를 기반으로 그래프를 생성할 수 있습니다. 장면에서 3D 노드를 그래프로 표시하는 애플리케이션을 생성하려면 먼저 3D 노드를 워크스페이스의 개체에 바인딩합니다. 이 매핑을 사용하면는 장면에 있는 3D 모델과 워크스페이스의 개체 간의 관계를 AWS IoT TwinMaker 그래프로 표시합니다. 그런 다

음 웹 애플리케이션을 생성하고, 장면으로 3D 모델을 선택하고, 그래프 형식으로 다른 개체와의 관계를 탐색할 수 있습니다.



AWS IoT 앱 키트 구성 요소를 활용하여 AWS IoT TwinMaker 장면에서 그래프를 생성하는 작업 웹 애플리케이션의 예는 github의 AWS IoT TwinMaker 샘플 반응 앱을 참조하세요.

AWS IoT TwinMaker 장면 그래프 사전 조건

장면에서 AWS IoT TwinMaker 지식 그래프를 사용하는 웹 앱을 생성하기 전에 다음 사전 조건을 완료하세요.

- AWS IoT TwinMaker 워크스페이스를 생성합니다. AWS IoT TwinMaker 콘솔에서 작업 영역을 생성할 수 있습니다.
- AWS IoT TwinMaker의 개체 구성 요소 시스템과 개체를 생성하는 방법을 숙지합니다. 자세한 내용은 첫 번째 개체 생성 단원을 참조하십시오.
- 3D 모델로 채워진 AWS IoT TwinMaker 장면을 생성합니다.
- AWS IoT TwinMaker의 AWS IoT 앱 키트 구성 요소를 숙지합니다. AWS IoT TwinMaker 구성 요소에 대한 자세한 내용은 섹션을 참조하세요AWS IoT TwinMaker UI 컴포넌트를 사용하여 사용자 지정 웹 애플리케이션 만들기.
- 지식 그래프 개념과 주요 용어를 숙지하십시오. AWS IoT TwinMaker 지식 그래프 핵심 개념을(를) 참조하세요.

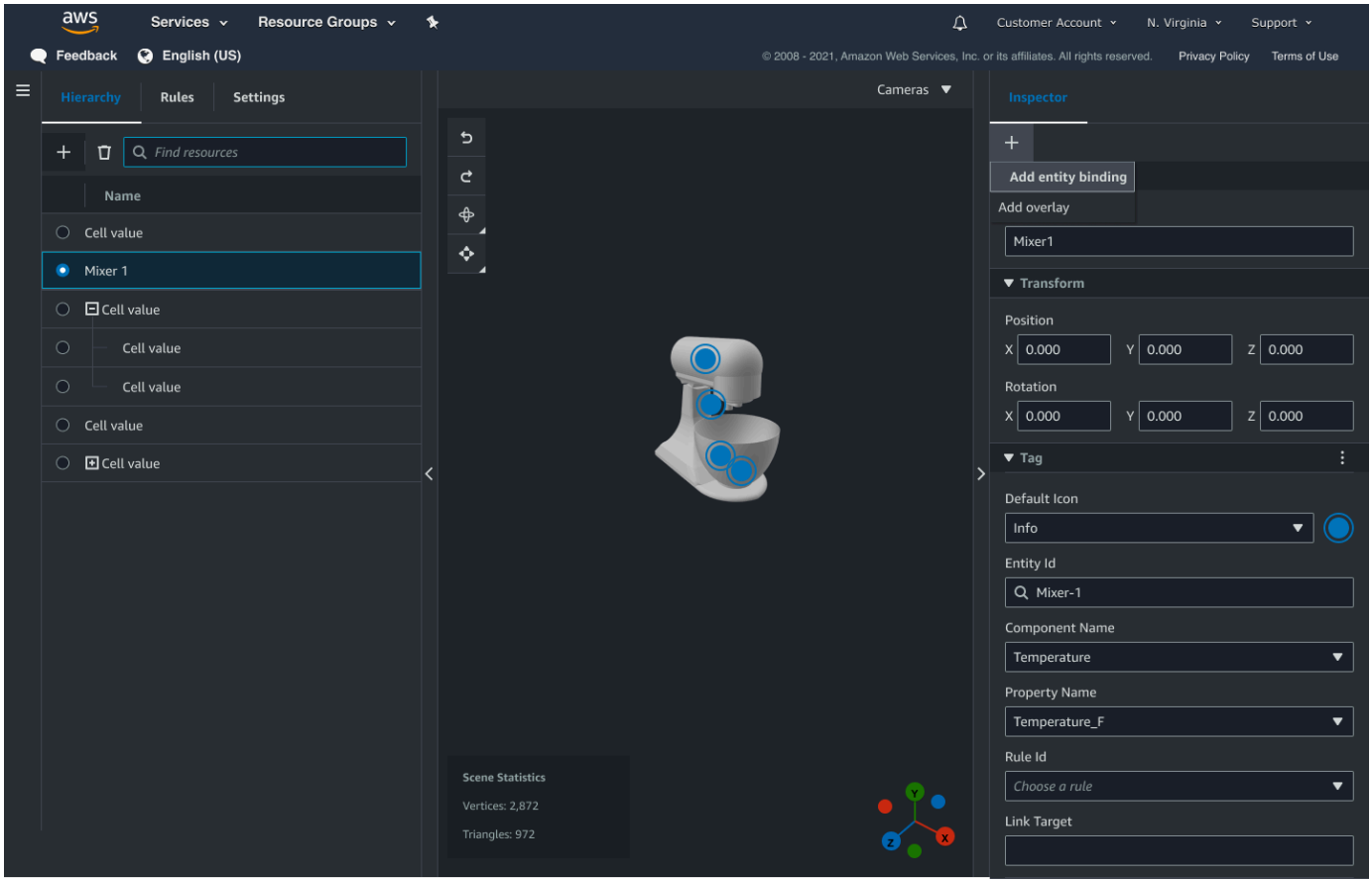
> **Note**
>
> AWS IoT TwinMaker 지식 그래프 및 관련 기능을 사용하려면 표준 또는 계층형 번들 요금 모드에 있어야 합니다. AWS IoT TwinMaker 요금에 대한 자세한 내용은 섹션을 참조하세요AWS IoT TwinMaker 가격 책정 모드 전환.

장면에서 3D 노드를 바인딩하기

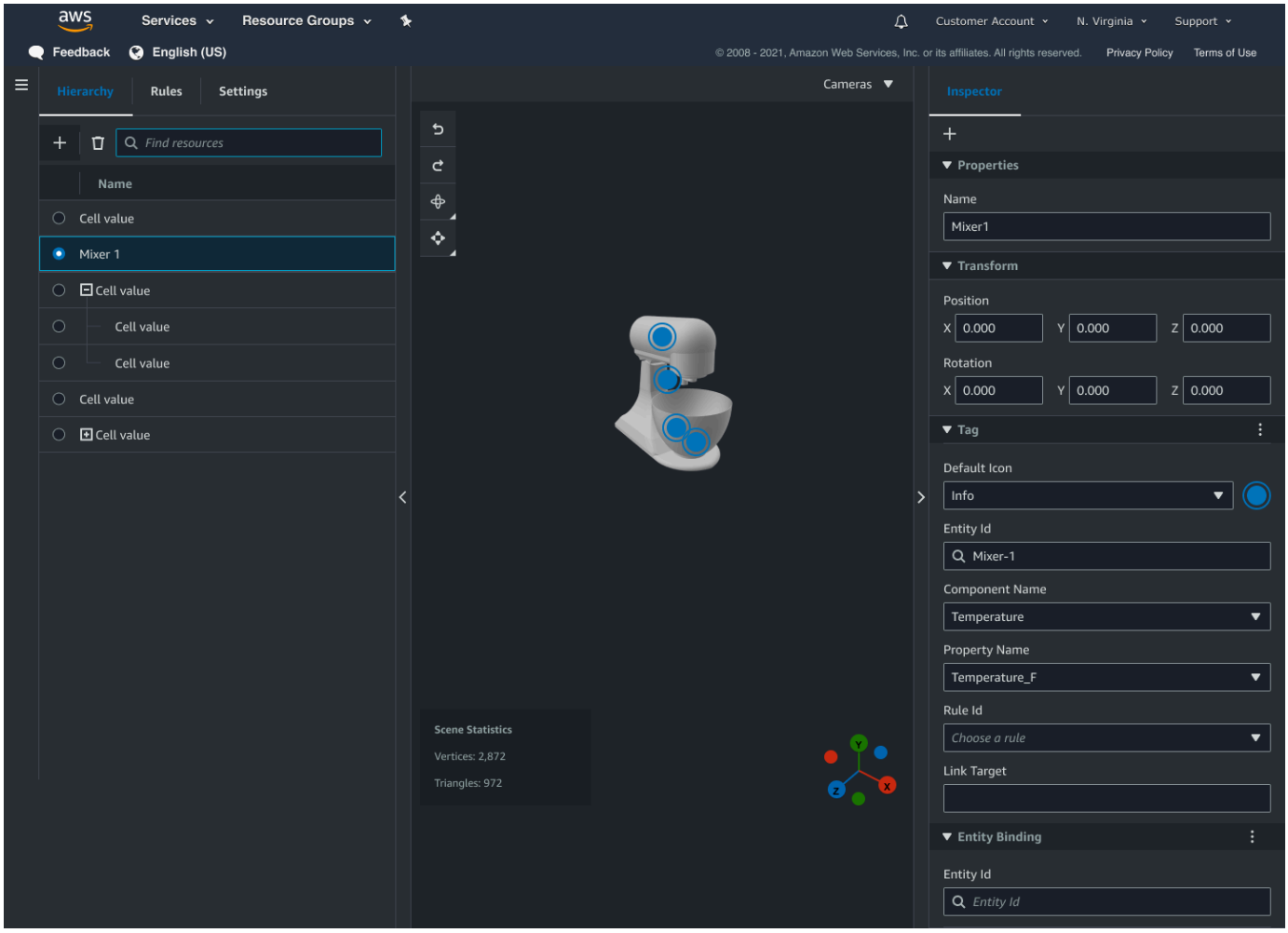
지식 그래프를 장면과 통합하는 웹 앱을 생성하기 전에 장면에 있는 3D 노드라고 하는 3D 모델을 연결된 워크스페이스 개체에 바인딩합니다. 예를 들어 장면에 믹서 장비 모델이 있고 라는 해당 개체가 있는 경우 믹서 모델과 믹서를 나타내는 개체 간에 데이터 바인딩을 `mixer_0`생성하여 모델과 개체를 그래프로 표시할 수 있습니다.

데이터 바인딩 작업을 수행하려면

1. AWS IoT TwinMaker 콘솔에 로그인합니다.
2. 작업 영역을 열고 바인딩하려는 3D 노드가 있는 장면을 선택합니다.
3. 장면 컴포저에서 노드(3D 모델)를 선택합니다. 노드를 선택하면 화면 오른쪽에 검사기 패널이 열립니다.
4. 검사기 패널에서 패널 상단으로 이동하여 + 버튼을 선택합니다. 그런 다음 개체 바인딩 추가 옵션을 선택합니다. 그러면 현재 선택한 노드에 바인딩할 개체를 선택할 수 있는 드롭다운이 열립니다.



5. 데이터 바인딩 드롭다운 메뉴에서 3D 모델에 매핑할 개체 ID를 선택합니다. 구성 요소 이름 및 속성 이름 필드에서 바인딩하려는 구성 요소 및 속성을 선택합니다.



일단 개체 ID, 구성 요소 이름 및 속성 이름 필드를 선택하면 바인딩이 완료됩니다.

6. 그래프로 표시하려는 모든 모델 및 개체에 대해 이 프로세스를 반복합니다.

> **Note**
>
> 장면 태그에서도 동일한 데이터 바인딩 작업을 수행할 수 있습니다. 간단하게 개체 대신 태그를 선택하고 동일한 프로세스에 따라 태그를 노드에 바인딩하기만 하면 됩니다.

웹 애플리케이션 생성

엔터티를 바인딩한 후 AWS IoT 앱 키트 라이브러리를 사용하여 장면을 보고 장면 노드와 엔터티 간의 관계를 탐색할 수 있는 지식 그래프 위젯이 있는 웹 앱을 빌드합니다.

다음 리소스를 사용하여 자신만의 앱을 만들어 보십시오.

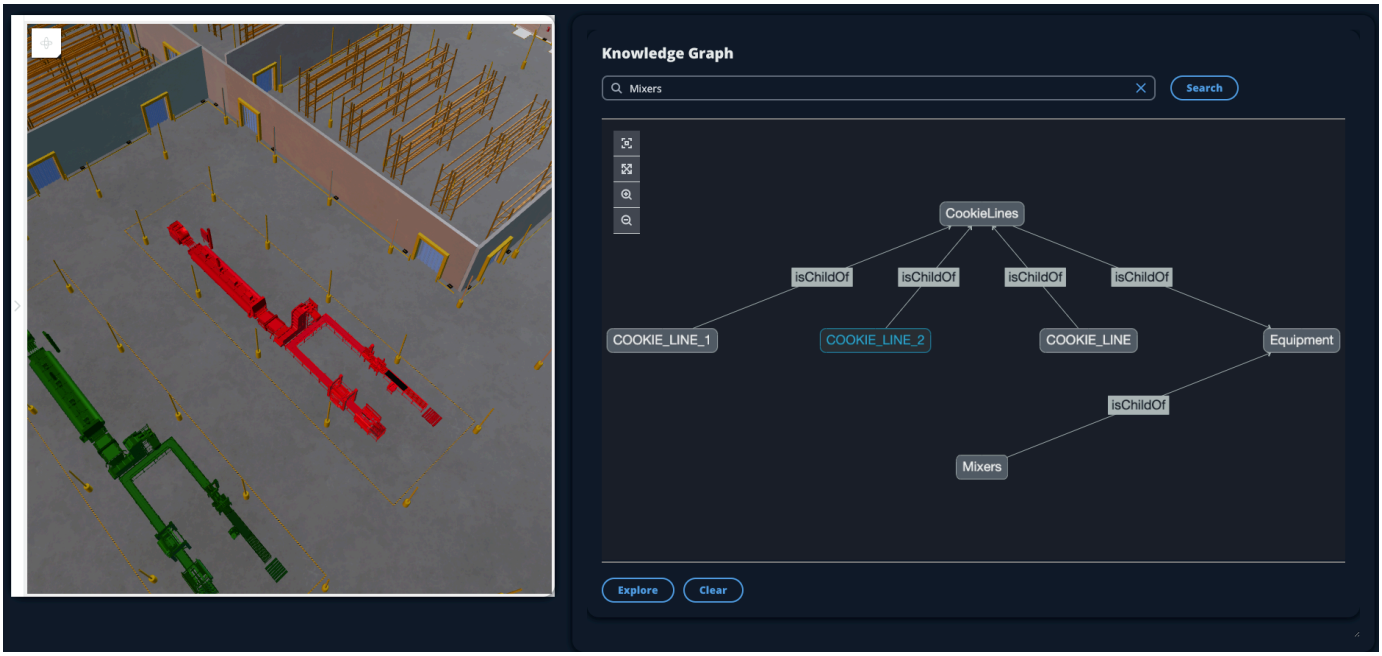
- AWS IoT TwinMaker 샘플 반응 앱 github Readme 설명서.
- github의 AWS IoT TwinMaker 샘플 반응 앱 소스입니다.
- AWS IoT 앱 키트 시작하기 설명서.
- AWS IoT 앱 키트 Video Player 구성 요소 설명서.
- AWS IoT 앱 키트 장면 뷰어 구성 요소 설명서.

다음 절차에서는 웹 앱에서 장면 뷰어 구성 요소의 기능을 보여줍니다.

> **Note**
>
> 이 절차는 AWS IoT TwinMaker 샘플 반응 AWS IoT 앱에서 앱 키트 장면 뷰어 구성 요소의 구현을 기반으로 합니다.

1. AWS IoT TwinMaker 샘플 반응 앱의 장면 뷰어 구성 요소를 엽니다. 검색 필드에 엔터티 이름 또는 부분 엔터티 이름(대소문자 구분 검색)을 입력한 다음 검색 버튼을 선택합니다. 모델이 개체 ID에 바인딩되면 장면의 모델이 강조 표시되고 개체의 노드가 장면 뷰어 패널에 표시됩니다.

2. 모든 관계의 그래프를 생성하려면 장면 뷰어 위젯에서 노드를 선택하고 탐색 버튼을 선택합니다.



3. 지우기 버튼을 눌러 현재 그래프 선택을 지우고 다시 시작합니다.

Grafana에서 AWS IoT TwinMaker 지식 그래프를 사용하는 방법

이 섹션에서는 AWS IoT TwinMaker Grafana 대시보드에 쿼리 편집기 패널을 추가하여 쿼리를 실행하고 표시하는 방법을 보여줍니다.

AWS IoT TwinMaker 쿼리 편집기 사전 조건

Grafana에서 AWS IoT TwinMaker 지식 그래프를 사용하기 전에 다음 사전 조건을 완료하세요.

- AWS IoT TwinMaker 워크스페이스를 생성합니다. AWS IoT TwinMaker 콘솔에서 작업 영역을 생성할 수 있습니다.
- Grafana와 함께 사용하도록 AWS IoT TwinMaker 를 구성합니다. 지침은 AWS IoT TwinMaker Grafana 대시보드 통합 섹션을 참조하세요.

> **Note**
>
> AWS IoT TwinMaker 지식 그래프를 사용하려면 표준 또는 계층형 번들 요금 모드에 있어야 합니다. 자세한 내용은 AWS IoT TwinMaker 가격 책정 모드 전환 단원을 참조하십시오.

AWS IoT TwinMaker 쿼리 편집기 권한

Grafana에서 AWS IoT TwinMaker 쿼리 편집기를 사용하려면 작업에 대한 권한이 있는 IAM 역할이 있어야 합니다`iottwinmaker:ExecuteQuery`. 이 예제와 같이 워크스페이스 대시보드 역할에 해당 권한을 추가합니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iottwinmaker:GetEntity",
                "iottwinmaker:ListEntities",
                "iottwinmaker:ExecuteQuery"
            ],
            "Resource": [
                "arn:aws:iottwinmaker:us-
east-2:111122223333:workspace/workspaceId",
                "arn:aws:iottwinmaker:us-
east-2:111122223333:workspace/workspaceId/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "iottwinmaker:ListWorkspaces",
            "Resource": "*"
        }
    ]
}
```

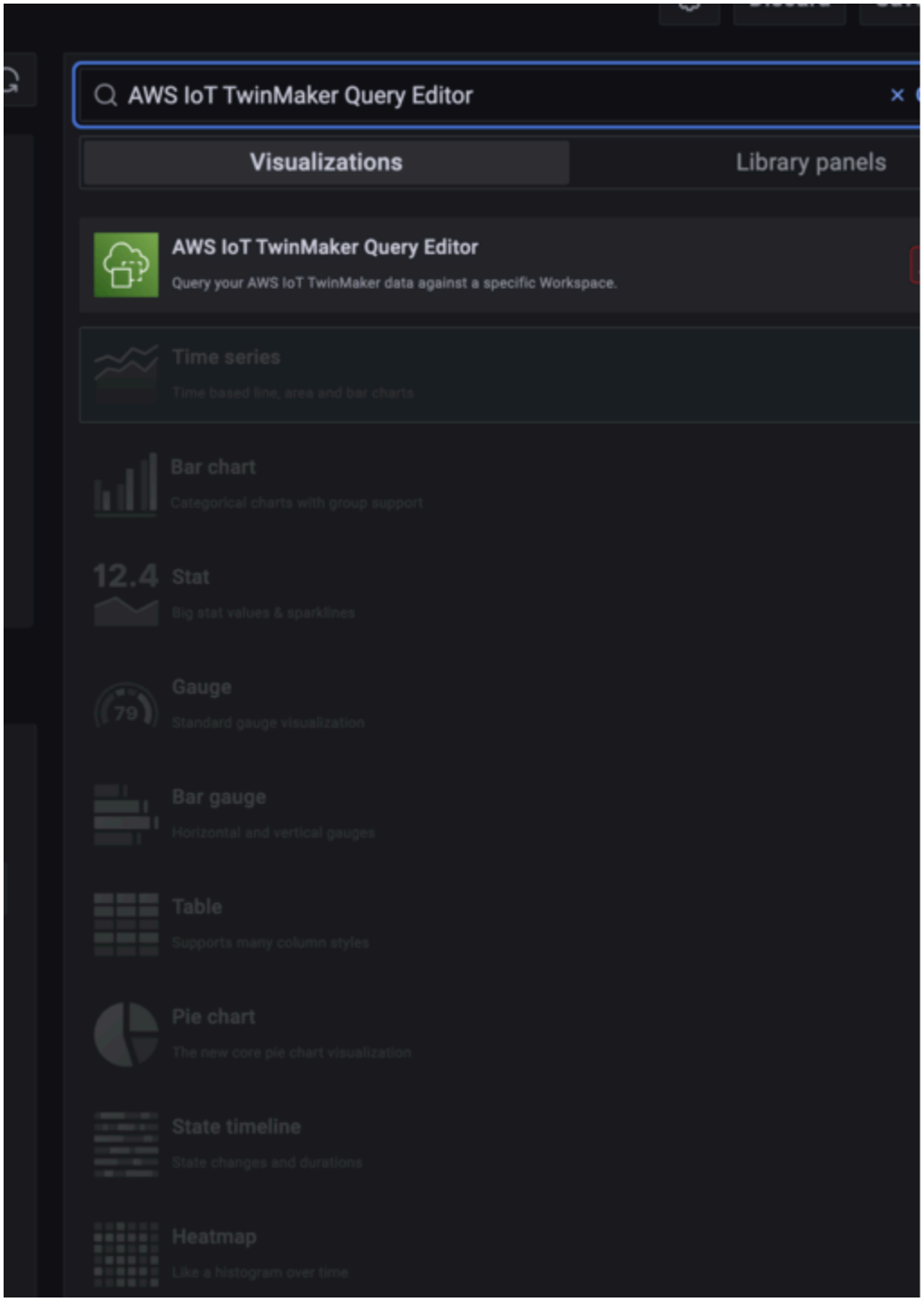
> **Note**
>
> AWS IoT TwinMaker Grafana 데이터 소스를 구성할 때 역할 ARN 수임 필드에이 권한이 있는 역할을 사용해야 합니다. 추가한 후에는 작업 영역 옆의 드롭다운에서 작업 영역을 선택할 수 있습니다.

자세한 내용은 대시보드 IAM 역할 생성 단원을 참조하십시오.

AWS IoT TwinMaker 쿼리 편집기 패널 설정

지식 그래프에 대한 새 Grafana 대시보드 패널을 설정하려면

1. AWS IoT TwinMaker Grafana 대시보드를 엽니다.
2. 새 대시보드 패널을 만드십시오. 패널을 생성하는 방법에 대한 자세한 단계는 Grafana 설명서의 대시보드 생성을 참조하세요.
3. 시각화 목록에서 AWS IoT TwinMaker 쿼리 편집기를 선택합니다.



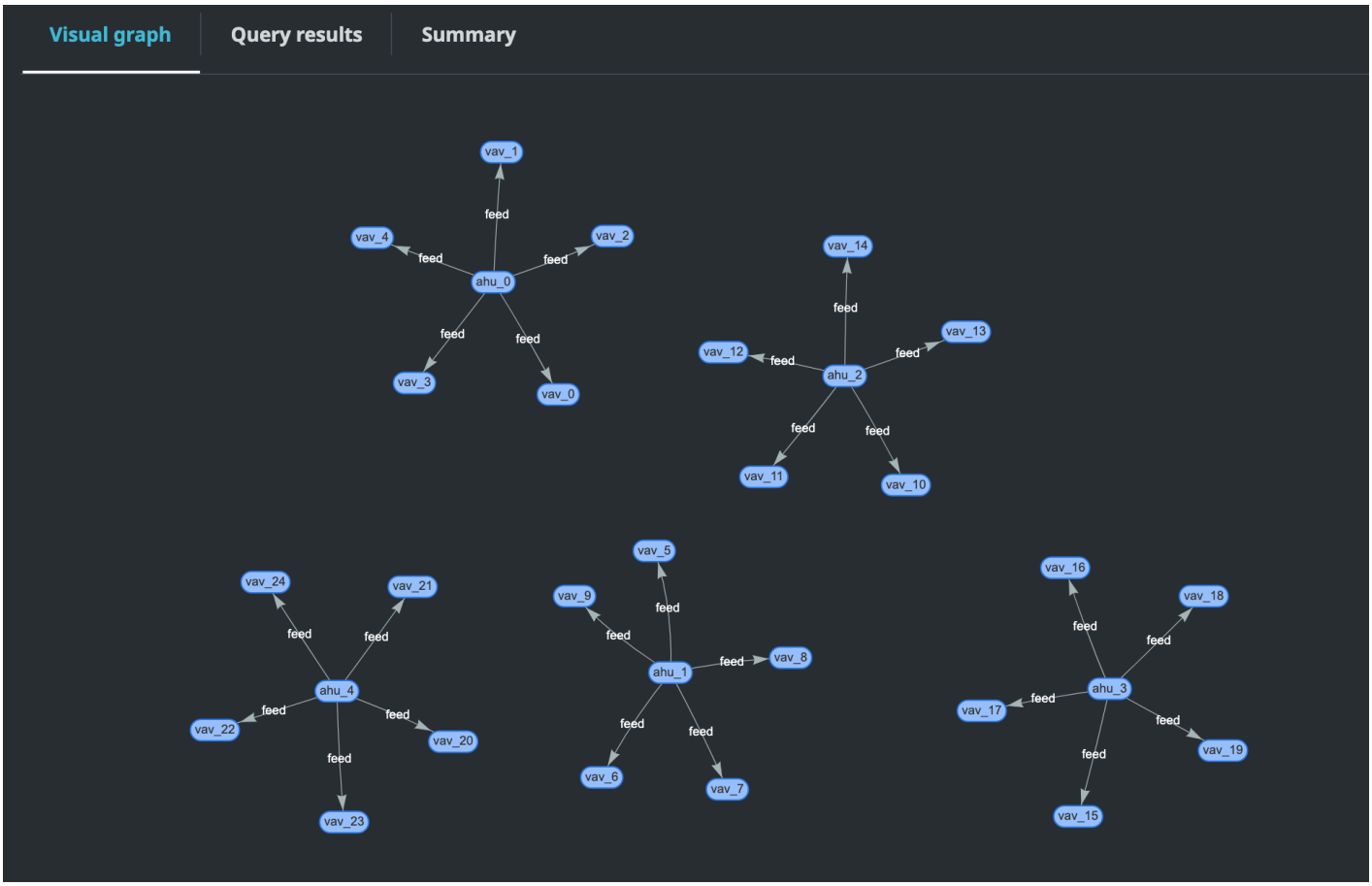
4. 데이터 소스를 선택하여 쿼리를 실행합니다.
5. (선택 사항) 제공된 필드에 새 패널의 이름을 추가합니다.
6. 적용을 선택하여 새 패널을 저장하고 확인합니다.

지식 그래프 패널은 AWS IoT TwinMaker 콘솔에 제공된 쿼리 편집기와 유사한 방식으로 작동합니다. 패널에서 작성한 쿼리를 실행하고, 작성하고, 지울 수 있습니다. 쿼리를 작성하는 방법에 대한 자세한 내용은 섹션을 참조하세요AWS IoT TwinMaker 지식 그래프 추가 리소스.

AWS IoT TwinMaker 쿼리 편집기 사용 방법

쿼리 결과는 다음 이미지와 같이 세 가지 방식으로 표시됩니다. 그래프로 시각화하거나, 표로 나열하거나, 실행 요약으로 표시됩니다.

- 그래프 시각화:



  시각적 그래프에는 결과에 적어도 하나 이상의 관계가 있는 쿼리에 대한 데이터만 표시됩니다. 그래프에는 개체가 노드로 표시되고 관계는 그래프의 방향 엣지로 표시됩니다.

- 테이블 형식 데이터:

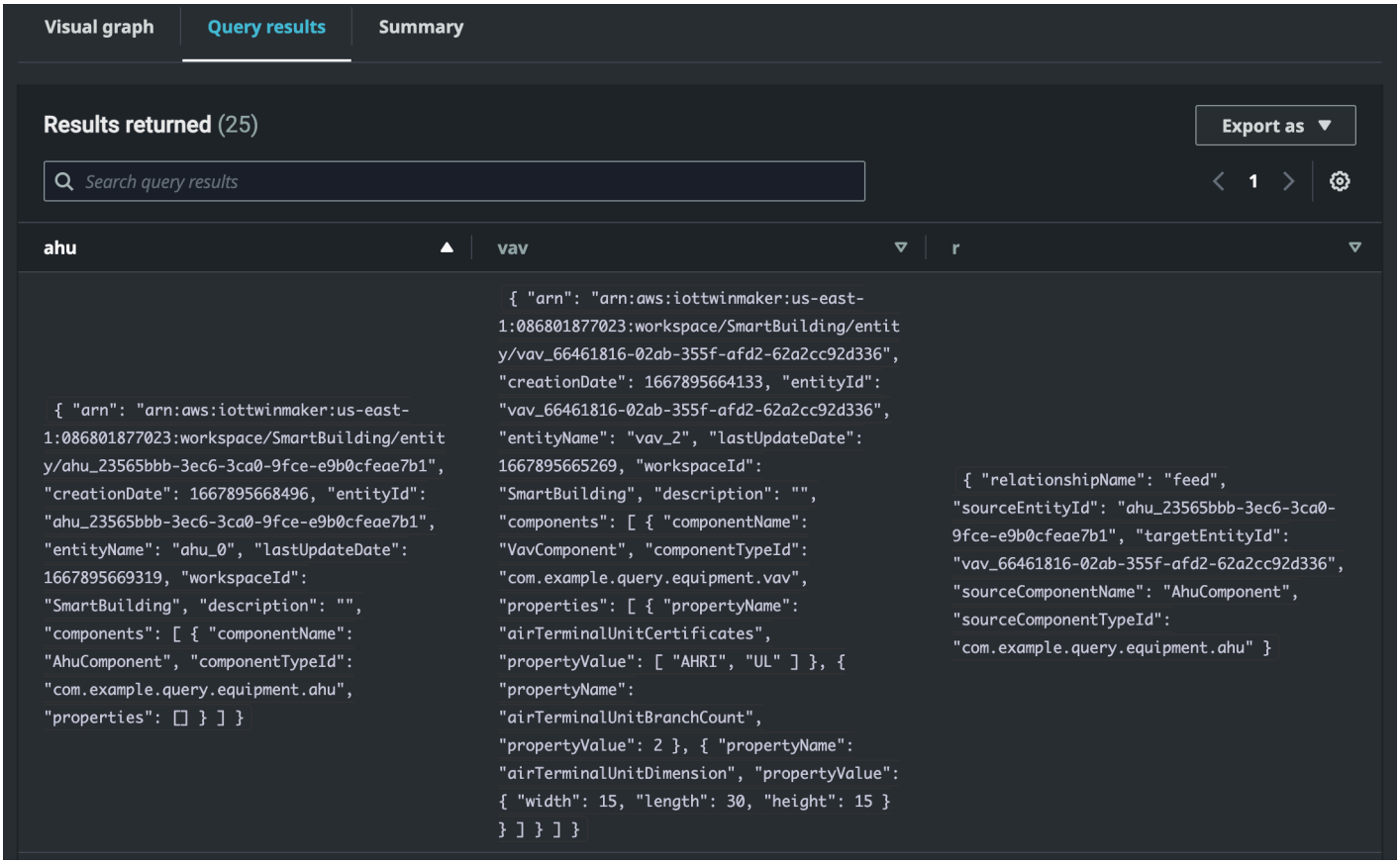
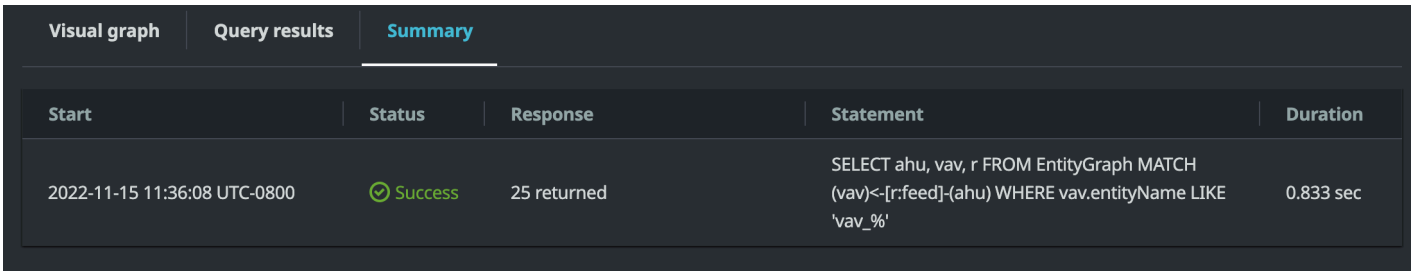


표 형식 데이터 형식은 모든 쿼리의 데이터를 표시합니다. 테이블에서 특정 결과 또는 결과의 하위 집합을 검색할 수 있습니다. 데이터는 JSON 또는 CSV 형식으로 내보낼 수 있습니다.

- 규칙 요약



실행 요약에는 쿼리 상태에 대한 쿼리와 메타데이터가 표시됩니다.

AWS IoT TwinMaker 지식 그래프 추가 리소스

이 섹션에서는 지식 그래프에 쿼리를 작성하는 데 사용되는 PartiQL 구문의 기본 예제와 지식 그래프 데이터 모델에 대한 정보를 제공하는 PartiQL 설명서 링크를 제공합니다.

- PartiQL 그래프 데이터 모델 문서
- PartiQL 그래프 쿼리 문서

이 예제 세트는 응답이 포함된 기본 쿼리를 보여줍니다. 이를 참조로 사용하여 자체 쿼리를 작성합니다.

기본 쿼리

- 필터가 있는 모든 개체 얻기

```
SELECT entity
FROM EntityGraph MATCH (entity)
WHERE entity.entityName = 'room_0'
```

이 쿼리는 이름이 인 워크스페이스의 모든 엔터티를 반환합니다`room_0`.

`FROM` 절: `EntityGraph`는 워크스페이스의 모든 엔터티와 해당 관계를 포함하는 그래프 컬렉션입니다. 이 컬렉션은 워크스페이스의 엔터티를 AWS IoT TwinMaker 기반으로에서 자동으로 생성되고 관리됩니다.

`MATCH`절: 그래프의 일부와 일치하는 패턴을 지정합니다. 이 경우, 패턴은 그래프의 모든 노드와 `(entity)` 일치하며 개체 변수에 바인딩됩니다. `FROM`절 뒤에 `MATCH`절이 와야 합니다.

`WHERE` 절: 값이와 일치해야 하는 노드의 `entityName` 필드에 필터를 지정합니다`room_0`.

`SELECT` 절: 전체 개체 노드가 반환되도록 `entity` 변수를 지정합니다.

응답:

```
{
  "columnDescriptions": [
    {
      "name": "entity",
      "type": "NODE"
    }
  ],
  "rows": [
    {
      "rowData": [
        {
```

```
          "arn": "arn:aws:iottwinmaker:us-east-1: 577476956029: workspace /
 SmartBuilding8292022 / entity / room_18f3ef90 - 7197 - 53 d1 - abab -
 db9c9ad02781 ",
          "creationDate": 1661811123914,
          "entityId": "room_18f3ef90-7197-53d1-abab-db9c9ad02781",
          "entityName": "room_0",
          "lastUpdateDate": 1661811125072,
          "workspaceId": "SmartBuilding8292022",
          "description": "",
          "components": [
            {
              "componentName": "RoomComponent",
              "componentTypeId": "com.example.query.construction.room",
              "properties": [
                {
                  "propertyName": "roomFunction",
                  "propertyValue": "meeting"
                },
                {
                  "propertyName": "roomNumber",
                  "propertyValue": 0
                }
              ]
            }
          ]
        }
      ]
    }
  ]
}
```


는 이름 및 유형과 같은 열에 대한 메타데이터를 columnDescriptions 반환합니다. 반환되는 유형은 NODE입니다. 이는 전체 노드가 반환되었음을 나타냅니다. 유형의 다른 값은 관계를 나타내EDGE거나 정수 또는 문자열과 같은 스칼라 값을 VALUE 나타낼 수 있습니다.

rows은 행 목록을 반환합니다. 오직 하나의 개체만 일치하므로, 하나인 은하나뿐이므로 개체 안의 모든 필드를 포함하는 하나의 rowData이(가) 반환됩니다.

> **Note**
>
> 스칼라 값만 반환할 수 있는 SQL과는 달리, PartiQL을 사용하여 객체(JSON으로서)를 반환할 수 있습니다.

각 노드에는 entityId, arn , 등의 모든 개체 수준 필드components, componentName, 등의 구성 요소 수준 필드properties, componentTypeId 및 propertyName 등의 속성 수준 필드가 모두 중첩 JSONpropertyValue으로 포함됩니다.

- 필터를 가진 모든 관계 얻기:

```
SELECT relationship
FROM EntityGraph MATCH (e1)-[relationship]->(e2)
WHERE relationship.relationshipName = 'isLocationOf'
```

이 쿼리는 관계 이름 isLocationOf을(를) 가진 작업 영역의 모든 관계를 반환합니다.

MATCH 절: 방향이 지정된 엣지(로 표시됨( ))에 의해 연결되고 라는 변수에 바인딩되는 두 노드(로 표시됨-[]->)와 일치하는 패턴을 지정합니다relationship.

WHERE 절: 엣지의 relationshipName 필드에 필터를 지정합니다. 여기서 값은 입니다isLocationOf.

SELECT절: 전체 엣지 노드가 반환되도록 관계 변수를 지정합니다.

응답

```
{
    "columnDescriptions": [{
        "name": "relationship",
        "type": "EDGE"
    }],
    "rows": [{
        "rowData": [{
            "relationshipName": "isLocationOf",
            "sourceEntityId": "floor_83faea7a-ea3b-56b7-8e22-562f0cf90c5a",
            "targetEntityId": "building_4ec7f9e9-e67e-543f-9d1b- 235df7e3f6a8",
            "sourceComponentName": "FloorComponent",
            "sourceComponentTypeId": "com.example.query.construction.floor"
```

```
        }]
    },
        ... //rest of the rows are omitted
    ]
}
```

의 열 유형은 columnDescriptions입니다EDGE.

각각은와 같은 필드가 있는 엣지를 rowData 나타냅니다relationshipName. 이는 엔터티에 정의된 관계 속성 이름과 동일합니다. sourceEntityId, sourceComponentName 및는 관계 속성이 정의된 엔터티 및 구성 요소에 대한 정보를 sourceComponentTypeId 제공합니다. 는 이 관계가 가리키는 개체를 targetEntityId 지정합니다.

- 특정 엔터티와 특정 관계가 있는 모든 엔터티 가져오기

```
SELECT e2.entityName
      FROM EntityGraph MATCH (e1)-[r]->(e2)
      WHERE relationship.relationshipName = 'isLocationOf'
      AND e1.entityName = 'room_0'
```

이 쿼리는 엔터티와 isLocationOf 관계가 있는 모든 엔터티의 모든 room_0엔터티 이름을 반환합니다.

MATCH 절: 방향이 지정된 엣지(e2)가 있는 두 노드(e1,)와 일치하는 패턴을 지정합니다r.

WHERE절: 은 관계 이름 및 소스 개체 이름에 대한 필터를 지정합니다.

SELECT 절:는 e2 노드의 entityName 필드를 반환합니다.

응답

```
{
  "columnDescriptions": [
    {
       "name": "entityName",
       "type": "VALUE"
    }
  ],
   "rows": [
    {
       "rowData": [
         "floor_0"
```

```
      ]
    }
  ]
}
```

columnDescriptions에서 열의 유형은 입니다.는 문자열entityName이기 VALUE 때문입니다.

하나의 개체인 floor_0가 반환됩니다.

일치

MATCH 절에서 지원되는 패턴은 다음과 같습니다.

- 노드 'a'를 가리키는 노드 'b' 일치:

```
FROM EntityGraph MATCH (a)-[rel]-(b)
```

- 노드 'b'를 가리키는 노드 'a' 일치:

```
FROM EntityGraph MATCH (a)-[]->(b)
```

관계에 필터를 지정할 필요가 없다고 가정할 때 관계에 바인딩된 변수는 없습니다.

- 노드 'b'를 가리키는 노드 'a'와 노드 'a'를 가리키는 노드 'b'를 일치시킵니다.

```
FROM EntityGraph MATCH (a)-[rel]-(b)
```

이렇게 하면 두 개의 일치 항목이 반환됩니다. 하나는 'a'에서 'b'로, 다른 하나는 'b'에서 'a'로 반환되므로 가능하면 방향이 지정된 엣지를 사용하는 것이 좋습니다.

- 관계 이름은 속성 그래프의 레이블이기도 EntityGraph하므로 WHERE 절에 필터를 지정하는 대신 콜론(:) 뒤에 관계 이름을 지정하면 rel.relationshipName 됩니다.

```
FROM EntityGraph MATCH (a)-[:isLocationOf]-(b)
```

- 체인 연결: 패턴을 체인으로 연결하여 여러 관계에서 일치시킬 수 있습니다.

```
FROM EntityGraph MATCH (a)-[rel1]->(b)-[rel2]-(c)
```

- 가변 홉 패턴은 여러 노드와 엣지에 걸쳐 있을 수도 있습니다.

```
FROM EntityGraph MATCH (a)-[]->{1,5}(b)
```

이 쿼리는 홉 1~5개 내에서 노드 'a'의 발신 엣지가 있는 모든 패턴과 일치합니다. 허용 한정자는 다음과 같습니다.

`{m,n}`- m회와 n회 사이의 반복

`{m,}`- m 회 이상의 반복

발신:

개체 노드에는 속성과 같은 추가 중첩 데이터가 포함된 구성 요소와 같은 중첩 데이터가 포함될 수 있습니다. MATCH 패턴의 결과를 중첩 해제하여 이러한 데이터에 액세스할 수 있습니다.

```
SELECT e
FROM EntityGraph MATCH (e), e.components AS c, c.properties AS p
WHERE c.componentTypeId = 'com.example.query.construction.room',
AND p.propertyName = 'roomFunction'
AND p.propertyValue = 'meeting'
```

변수에 . 점을 찍어 중첩된 필드에 액세스할 수 있습니다. 쉼표(,)는 엔터티를 구성 요소 내부에서 중첩 해제(또는 조인)한 다음 해당 구성 요소 내부의 속성을 포함하는 데 사용됩니다. AS는 WHERE 또는 SELECT 절에서 사용할 수 있도록 변수를 중첩되지 않은 변수에 바인딩하는 데 사용됩니다. 이 쿼리는 구성 요소 유형이 id `com.example.query.construction.room`인 구성 요소에서 `meeting`값을 사용하여 `roomFunction` 이름이 지정된 속성을 포함하는 모든 개체를 반환합니다.

개체에서 여러 구성 요소와 같은 필드의 여러 중첩된 필드에 액세스하려면 쉼표 표기법을 사용하여 결합을 수행하십시오.

```
SELECT e
FROM EntityGraph MATCH (e), e.components AS c1, e.components AS c2
```

선택:

- 노드 반환:

  ```
  SELECT e
  FROM EntityGraph MATCH (e)
  ```

- 엣지 반환:

  ```
  SELECT r
  FROM EntityGraph MATCH (e1)-[r]->(e2)
  ```

- 스칼라 값을 반환합니다.

```
SELECT floor.entityName, room.description, p.propertyValue AS roomfunction
FROM EntityGraph MATCH (floor)-[:isLocationOf]-(room),
room.components AS c, c.properties AS p
```

AS를 사용하는 별칭을 지정하여 출력 필드의 이름을 포맷합니다. 여기에서 응답의 열 이름 propertyValue대신에 roomfunction이(가) 반환됩니다.

- 별칭 반환:

```
SELECT floor.entityName AS floorName, luminaire.entityName as luminaireName
FROM EntityGraph MATCH (floor)-[:isLocationOf]-(room)-[:hasPart]-
(lightingZone)-[:feed]-(luminaire)
WHERE floor.entityName = 'floor_0'
AND luminaire.entityName like 'lumin%'
```

별칭을 사용하는 것은 명시적이고, 가독성을 높이고, 쿼리의 모호성을 방지하는 것이 좋습니다.

WHERE:

- 지원되는 논리 연산자는 AND, NOT및 입니다OR.
- 지원된 비교 연산자는 e <, <=, >, =>, = 및 !=입니다.
- 동일한 필드에 여러 OR 조건을 지정하려면 IN 키워드를 사용합니다.
- 개체, 구성 요소 또는 속성 필드에 필터링:

```
FROM EntityGraph MATCH (e), e.components AS c, c.properties AS p
WHERE e.entityName = 'room_0'
AND c.componentTypeId = 'com.example.query.construction.room',
AND p.propertyName = 'roomFunction'
AND NOT p.propertyValue = 'meeting'
OR p.propertyValue = 'office'
```

- configuration 속성을 기준으로 필터링합니다. 다음은 구성 맵의 키이고 unit는 값Celsius입니다.

```
WHERE p.definition.configuration.unit = 'Celsius'
```

- 맵 속성에 지정된 키와 값이 포함되어 있는지 확인하십시오:

```
WHERE p.propertyValue.length = 20.0
```

- 맵 속성에 지정된 키가 포함되어 있는지 확인하십시오:

```
WHERE NOT p.propertyValue.length IS MISSING
```

- 목록 속성에 지정된 값이 포함되어 있는지 확인하십시오.

```
WHERE 10.0 IN p.propertyValue
```

- 이 lower() 함수는 대소문자를 구분하지 않고 비교하는 데 사용합니다. 기본 값으로, 모든 비교는 대소문자를 구분합니다.

```
WHERE lower(p.propertyValue) = 'meeting'
```

LIKE:

필드의 정확한 값을 모르고 지정된 필드에서 전체 텍스트 검색을 수행할 수 있는 경우에 유용합니다. %은 0회 이상을 나타냅니다.

```
WHERE e.entityName LIKE '%room%'
```

- 삽입사 검색: %room%
- 접두사 검색: room%
- 접미사 검색: %room
- 값에 '%'가 있는 경우 이스케이프 문자를에 LIKE 넣고 이스케이프 문자를 로 지정합니다ESCAPE.

```
WHERE e.entityName LIKE 'room\%' ESCAPE '\'
```

DISTINCT:

```
SELECT DISTINCT c.componentTypeId
FROM EntityGraph MATCH (e), e.components AS c
```

- DISTINCT 키워드는 최종 결과에서 중복된 항목을 제거합니다.

  DISTINCT는 복잡한 데이터 유형에는 지원되지 않습니다.

COUNT

```
SELECT COUNT(e), COUNT(c.componentTypeId)
```

```
FROM EntityGraph MATCH (e), e.components AS c
```

- COUNT 키워드는 쿼리 결과의 항목 수를 계산합니다.
- COUNT는 중첩된 복합 필드 및 그래프 패턴 필드에서 지원되지 않습니다.
- COUNT DISTINCT 및 중첩 쿼리에서는 집계가 지원되지 않습니다.

  예를 들어 COUNT(DISTINCT e.entityId)은 지원되지 않습니다.

경로

경로 프로젝션을 사용하여 쿼리할 때 다음 패턴 프로젝션이 지원됩니다.

- 변수 홉 쿼리

  ```
  SELECT p FROM EntityGraph MATCH p = (a)-[]->{1, 3}(b)
  ```

  이 쿼리는 1~3개의 홉 내에서 노드 에서 발신 엣지가 있는 모든 패턴의 노드 메타데이터를 일치시키고 프로젝션합니다.

- 홉 쿼리 수정

  ```
  SELECT p FROM EntityGraph MATCH p = (a)-[]->(b)<-[]-(c)
  ```

  이 쿼리는 엔터티 및 수신 엣지의 메타데이터를 일치시키고 b로 프로젝션합니다.

- 비지정 쿼리

  ```
  SELECT p FROM EntityGraph MATCH p = (a)-[]-(b)-[]-(c)
  ```

  이 쿼리는 b를 통해 a와 c를 연결하는 1개의 홉 패턴으로 노드의 메타데이터를 일치시키고 프로젝션합니다.

  ```
  {
      "columnDescriptions": [
          {
              "name": "path",
              "type": "PATH"
          }
      ],
      "rows": [
          {
              "rowData": [
  ```

```
                {
                    "path": [
                        {
                            "entityId": "a",
                            "entityName": "a"
                        },
                        {
                            "relationshipName": "a-to-b-relation",
                            "sourceEntityId": "a",
                            "targetEntityId": "b"
                        },
                        {
                            "entityId": "b",
                            "entityName": "b"
                        }
                    ]
                }
            ]
        },
        {
            "rowData": [
                {
                    "path": [
                        {
                            "entityId": "b",
                            "entityName": "b"
                        },
                        {
                            "relationshipName": "b-to-c-relation",
                            "sourceEntityId": "b",
                            "targetEntityId": "c"
                        },
                        {
                            "entityId": "c",
                            "entityName": "c"
                        }
                    ]
                }
            ]
        }
    ]
}
```

이 PATH 쿼리 응답은 b를 통해 a와 c 사이의 각 경로/패턴의 모든 노드와 엣지를 식별하는 메타데이터로만 구성됩니다.

LIMIT 및 OFFSET:

```
SELECT e.entityName
FROM EntityGraph MATCH (e)
WHERE e.entityName LIKE 'room_%'
LIMIT 10
OFFSET 5
```

LIMIT은(는) 쿼리에서 반환되는 결과 수를 지정하고 OFFSET은(는) 건너뛸 결과 수를 지정합니다.

LIMIT 및 maxResults:

다음 예제는 총 500개의 결과를 반환하지만 API 호출당 한 번에 50개만 표시하는 쿼리를 보여줍니다. 이 패턴은 예를 들어 UI에 50개의 결과만 표시할 수 있는 경우 표시된 결과의 양을 제한해야 하는 경우에 사용할 수 있습니다.

```
aws iottwinmaker execute-query \
--workspace-id exampleWorkspace \
--query-statement "SELECT e FROM EntityGraph MATCH (e) LIMIT 500"\
--max-results 50
```

- LIMIT 키워드는 쿼리에 영향을 미치고 결과 행을 제한합니다. 반환된 총 결과 수를 제한하지 않고 API 호출당 반환되는 결과 수를 제어해야 하는 경우를 사용합니다LIMIT.
- max-results는 ExecuteQuery API 작업의 선택적 파라미터입니다.는 API에만 적용되며 위 쿼리의 범위 내에서 결과를 읽는 방법max-results만 적용합니다.

  쿼리max-results에서를 사용하면 반환된 실제 결과 수를 제한하지 않고 표시되는 결과 수를 줄일 수 있습니다.

아래 쿼리는 결과의 다음 페이지를 반복합니다. 이 쿼리는 ExecuteQuery API 호출을 사용하여 51~100행을 반환합니다. 여기서 결과의 다음 페이지는에 의해 지정됩니다. next-token이 경우 토큰은 입니다"H7kyGmvK376L".

```
aws iottwinmaker execute-query \
--workspace-id exampleWorkspace \
--query-statement "SELECT e FROM EntityGraph MATCH (e) LIMIT 500"\
--max-results 50
```

```
--next-token "H7kyGmvK376L"
```

- `next-token` 문자열은 결과의 다음 페이지를 지정합니다. 자세한 내용은 ExecuteQuery API 작업을 참조하세요.

AWS IoT TwinMaker 지식 그래프 쿼리에는 다음과 같은 제한이 있습니다.

제한 이름	할당량	조정 가능
쿼리 실행 제한 시간	10초	아니요
최대 홉 수	10	예
최대 자체 수 `JOIN`	20	예
최대 보호되는 필드 수	20	예
최대 조건식 수(`AND`, `OR`, `NOT`)	10	예
`LIKE` 표현식 패턴의 최대 길이(와일드카드 및 이스케이프 포함)	20	예
`IN` 절에서 지정할 수 있는 최대 항목 수	10	예
의 최대값 `OFFSET`	3000	예
의 최대값 `LIMIT`	3000	예
최대 순회 값(`OFFSET` + `LIMIT`)	3000	예

와 자산 동기화 AWS IoT SiteWise

AWS IoT TwinMaker 는 자산 및 자산 모델에 대한 자산 동기화(자산 동기화)를 AWS IoT SiteWise 지원합니다. 자산 동기화는 AWS IoT SiteWise 구성 요소 유형을 사용하여 기존 AWS IoT SiteWise 자산 및 자산 모델을 가져와 이러한 리소스를 AWS IoT TwinMaker 개체, 구성 요소 및 구성 요소 유형으로 변환합니다. 다음 섹션에서는 자산 동기화를 구성하는 방법과 AWS IoT TwinMaker 워크스페이스에 동기화할 수 있는 AWS IoT SiteWise 자산 및 자산 모델을 안내합니다.

주제

와 자산 동기화 사용 AWS IoT SiteWise

이 주제에서는 AWS IoT SiteWise 자산 동기화를 켜고 구성하는 방법을 보여줍니다. 사용 중인 워크스페이스 유형에 따라 적절한 절차를 따릅니다.

> **Important**
>
> 사용자 지정 워크스페이스와 기본 워크스페이스의 차이점에 대한 the section called “사용자 지정 워크스페이스와 기본 워크스페이스의 차이점” 자세한 내용은 섹션을 참조하세요.

주제

사용자 지정 워크스페이스 사용

자산 동기화를 켜기 전에 이러한 사전 요구 사항을 검토하십시오.

사전 조건

사용하기 전에 AWS IoT SiteWise다음을 완료해야 합니다.

- AWS IoT TwinMaker 워크스페이스가 있습니다.
- 에 자산 및 자산 모델이 있습니다 AWS IoT SiteWise. 자세한 내용은 자산 모델 생성을 참조하십시오.
- 다음 AWS IoT SiteWise 작업에 대한 읽기 권한이 있는 기존 IAM 역할:
  - `ListAssets`
  - `ListAssetModels`
  - `DescribeAsset`
  - `DescribeAssetModel`
- IAM 역할에는에 대한 AWS IoT TwinMaker다음과 같은 쓰기 권한이 있어야 합니다.
  - `CreateEntity`
  - `UpdateEntity`
  - `DeleteEntity`
  - `CreateComponentType`
  - `UpdateComponentType`
  - `DeleteComponentType`
  - `ListEntities`
  - `GetEntity`
  - `ListComponentTypes`

  다음 IAM 역할을 필수 역할의 템플릿으로 사용합니다.

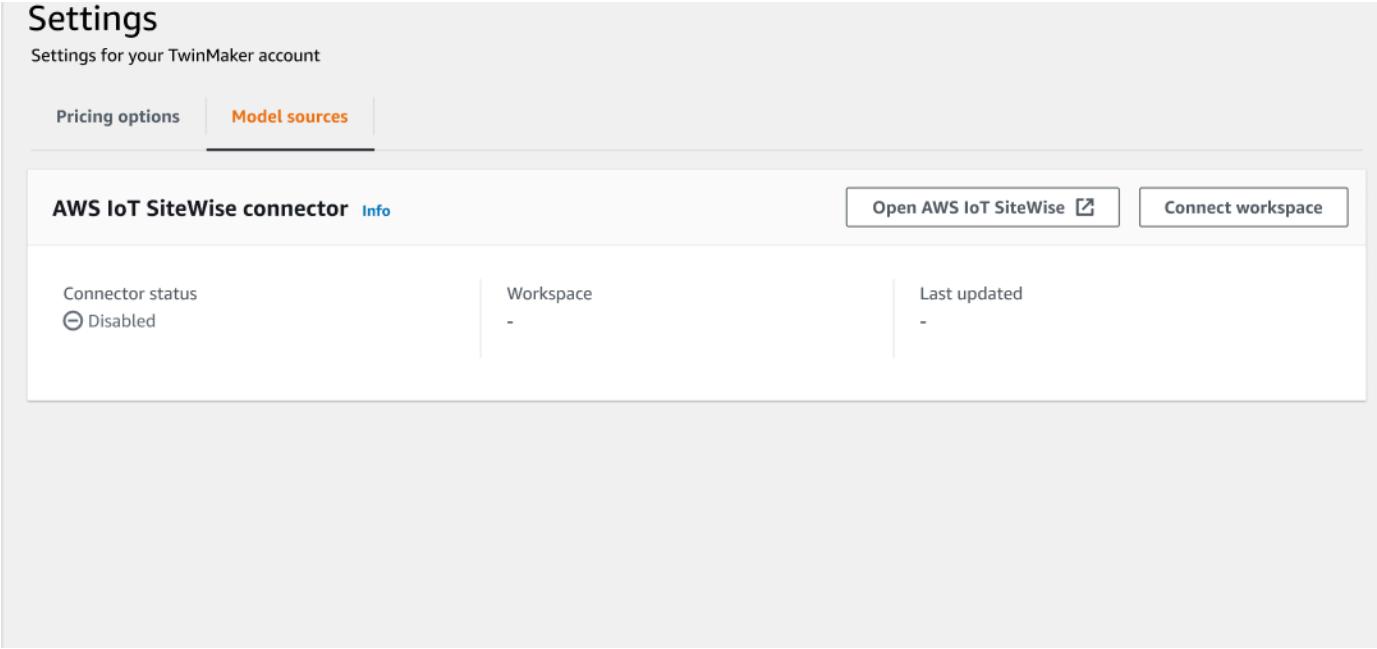
```
// trust relationships
 {
    {
        "Version": "2012-10-17",
        "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
            "Service": [
```

```
            "iottwinmaker.amazonaws.com"
            ]
        },
            "Action": "sts:AssumeRole"
        }
    ]
}

// permissions - replace ACCOUNT_ID, REGION, WORKSPACE_ID with actual values
{
    "Version": "2012-10-17",
    "Statement": [{
            "Sid": "SiteWiseAssetReadAccess",
            "Effect": "Allow",
            "Action": [
                "iotsitewise:DescribeAsset"
            ],
            "Resource": [
                "arn:aws:iotsitewise:REGION:ACCOUNT_ID:asset/*"
            ]
        },
        {
            "Sid": "SiteWiseAssetModelReadAccess",
            "Effect": "Allow",
            "Action": [
                "iotsitewise:DescribeAssetModel"
            ],
            "Resource": [
                "arn:aws:iotsitewise:REGION:ACCOUNT_ID:asset-model/*"
            ]
        },
        {
            "Sid": "SiteWiseAssetModelAndAssetListAccess",
            "Effect": "Allow",
            "Action": [
                "iotsitewise:ListAssets",
                "iotsitewise:ListAssetModels"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Sid": "TwinMakerAccess",
```

```
            "Effect": "Allow",
            "Action": [
                "iottwinmaker:GetEntity",
                "iottwinmaker:CreateEntity",
                "iottwinmaker:UpdateEntity",
                "iottwinmaker:DeleteEntity",
                "iottwinmaker:ListEntities",
                "iottwinmaker:GetComponentType",
                "iottwinmaker:CreateComponentType",
                "iottwinmaker:UpdateComponentType",
                "iottwinmaker:DeleteComponentType",
                "iottwinmaker:ListComponentTypes"
            ],
            "Resource": [
                "arn:aws:iottwinmaker:REGION:ACCOUNT_ID:workspace/WORKSPACE_ID",
                "arn:aws:iottwinmaker:REGION:ACCOUNT_ID:workspace/WORKSPACE_ID/*"

            ]
        }
    ]
}
```

다음 절차에 따라 AWS IoT SiteWise 자산 동기화를 켜고 구성합니다.

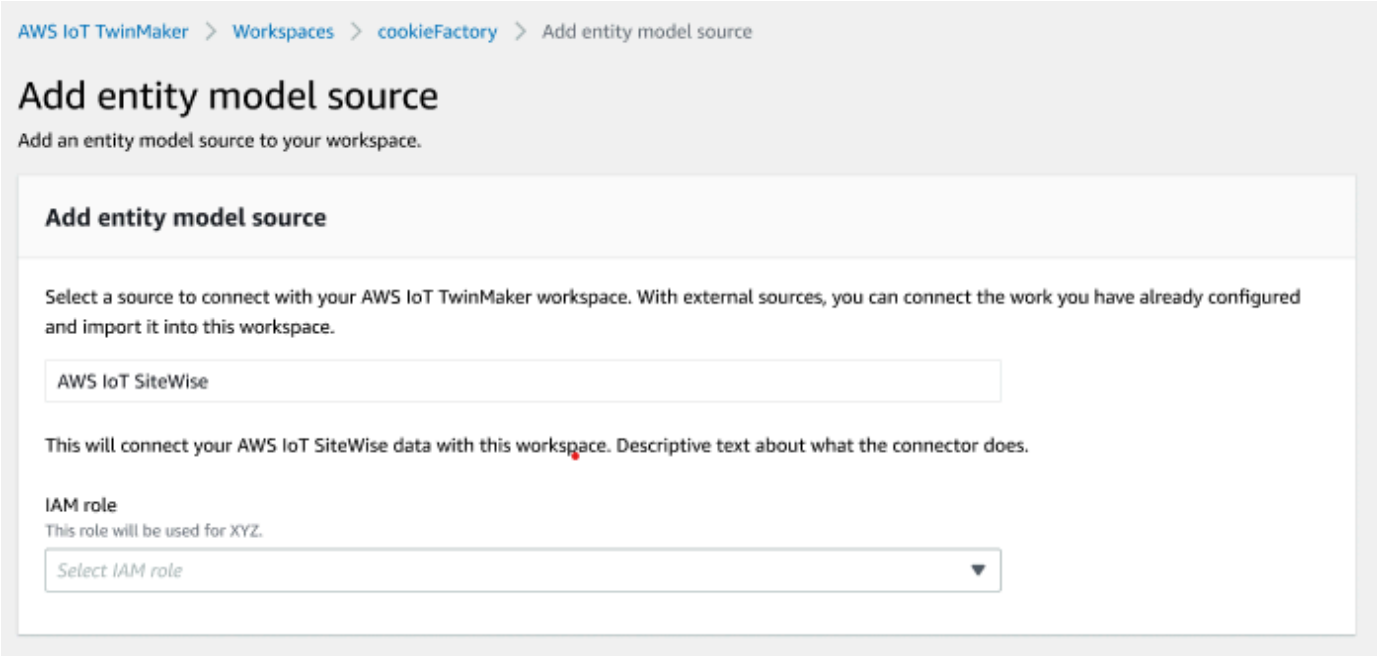
1. AWS IoT TwinMaker 콘솔에서, 설정페이지로 이동합니다.
2. 모델 소스 탭을 엽니다.



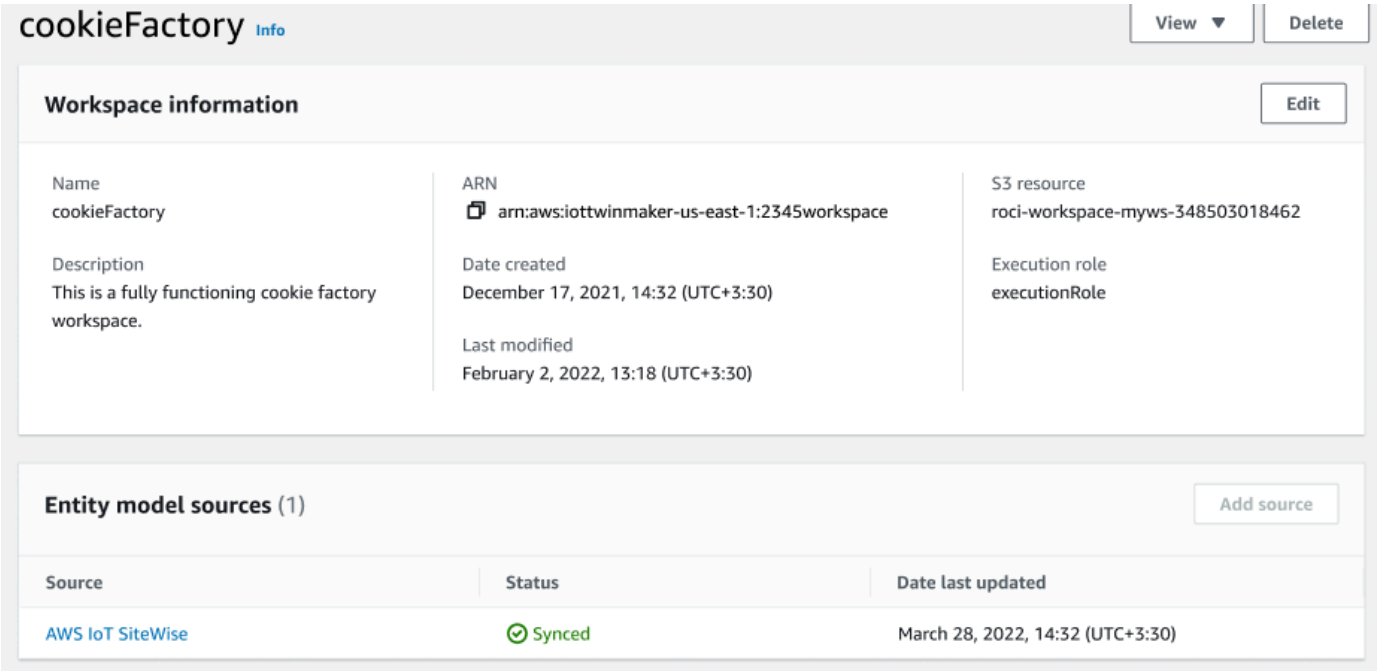
3. 워크스페이스 연결을 선택하여 워크스페이스를 AWS IoT SiteWise 자산에 연결합니다 AWS IoT TwinMaker .

> **Note**
>
> 단일 AWS IoT TwinMaker 워크스페이스에서만 자산 동기화를 사용할 수 있습니다. 각기 다른 작업 영역에서 동기화하려면 한 작업 영역에서 동기화 연결을 끊고 또다른 작업 영역에 연결해야 합니다.

4. 그런 다음 자산 동기화를 사용하려는 작업 영역으로 이동합니다.
5. 소스 추가를 선택합니다. 그러면 개체 모델 소스 추가 페이지가 열립니다.



6. 개체 모델 소스 추가 페이지에서 소스필드가 AWS IoT SiteWise을(를) 표시하는지 확인합니다. IAM 역할의 사전 요구 사항으로서 생성한 IAM 역할을 선택합니다.
7. 이제 AWS IoT SiteWise 자산 동기화가 활성화되었습니다. 선택한 워크스페이스 페이지 상단에 자산 동기화가 활성화되었음을 확인하는 확인 배너가 나타나는 것을 확인해야 합니다. 또한 이제 개체 모델 소스 섹션에 나열된 동기화 소스를 확인해야 합니다



IoTSiteWiseDefaultWorkspace 사용

AWS IoT SiteWiseAWS IoT TwinMaker 통합에 옵트인하면 라는 기본 워크스페이스IoTSiteWiseDefaultWorkspace가 생성되고와 자동으로 동기화됩니다 AWS IoT SiteWise.

API를 AWS IoT TwinMaker CreateWorkspace 사용하여 라는 워크스페이스를 생성할 수도 있습니다IoTSiteWiseDefaultWorkspace.

사전 조건

를 생성하기 전에 다음을 수행했는지 IoTSiteWiseDefaultWorkspace확인합니다.

- AWS IoT TwinMaker 서비스 연결 역할을 생성합니다. 자세한 정보는 에 서비스 연결 역할 사용 AWS IoT TwinMaker을 참조하세요.
- IAM 콘솔(https://console.aws.amazon.com/iam/)을 엽니다.

  역할 또는 사용자를 검토하고에 대한 권한이 있는지 확인합니다iotsitewise:EnableSiteWiseIntegration.

  필요한 경우 역할 또는 사용자에게 권한을 추가합니다.

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iotsitewise:EnableSiteWiseIntegration",
            "Resource": "*"
        }
    ]
}
```

사용자 지정 워크스페이스와 기본 워크스페이스의 차이점

> **Important**
>
> 와 같은 새로운 AWS IoT SiteWise 기능은 CompositionModel에서만 사용할 수 있습니다IoTSiteWiseDefaultWorkspace. 사용자 지정 워크스페이스 대신 기본 워크스페이스를 사용하는 것이 좋습니다.

를 사용할 때 자산 동기화와 함께 사용자 지정 워크스페이스를 사용하는 것과 몇 IoTSiteWiseDefaultWorkspace가지 주목할 만한 차이점이 있습니다.

- 기본 워크스페이스를 생성할 때 Amazon S3 위치 및 IAM 역할은 선택 사항입니다.

  > **Note**
  >
  > UpdateWorkspace를 사용하여 Amazon S3 위치 및 IAM 역할을 제공할 수 있습니다.

- 에는 리소스를 동기화할 AWS IoT SiteWise 리소스 수 제한이 IoTSiteWiseDefaultWorkspace 없습니다 AWS IoT TwinMaker.
- 에서 리소스를 동기화하면 AWS IoT SiteWise리소스SyncSource는가 됩니다SITEWISE_MANAGED. 여기에는 Entities 및가 포함됩니다ComponentTypes.
- 와 같은 새로운 AWS IoT SiteWise 기능은 CompositionModel에서만 사용할 수 있습니다IoTSiteWiseDefaultWorkspace.

에는 다음과 같은 몇 가지 제한 사항이 IoTSiteWiseDefaultWorkspace있습니다.

- 기본 워크스페이스는 삭제할 수 없습니다.
- 리소스를 삭제하려면 먼저 AWS IoT SiteWise 리소스를 삭제한 다음의 해당 리소스 AWS IoT TwinMaker 가 삭제됩니다.

에서 동기화된 리소스 AWS IoT SiteWise

이 주제에서는에서 AWS IoT TwinMaker 워크스페이스 AWS IoT SiteWise 로 동기화할 수 있는 자산을 나열합니다.

> **Important**
>
> 사용자 지정 워크스페이스와 기본 워크스페이스의 차이점에 대한 사용자 지정 워크스페이스와 기본 워크스페이스의 차이점 자세한 내용은 섹션을 참조하세요.

사용자 지정 및 기본 워크스페이스

다음 리소스는 동기화되어 사용자 지정 워크스페이스와 기본 워크스페이스 모두에서 사용할 수 있습니다.

자산 모델

AWS IoT TwinMaker 는의 각 자산 모델에 대해 새 구성 요소 유형을 생성합니다 AWS IoT SiteWise.

- `TypeId` 자산 모델의 구성 요소는 다음 패턴 중 하나를 사용합니다.
  - 사용자 지정 워크스페이스 - `iotsitewise.assetmodel:`*`assetModelId`*
  - 기본 워크스페이스 - *`assetModelId`*
- 자산 모델의 각 속성은 다음 이름 지정 패턴 중 하나를 사용하여 구성 요소 유형의 새 속성입니다.
  - 사용자 지정 워크스페이스 - `Property_`*`propertyId`*
  - 기본 워크스페이스 - *`propertyId`*

  의 속성 이름은 속성 정의`displayName`에 로 AWS IoT SiteWise 저장됩니다.
- 자산 모델의 각 계층 구조는 유형의 새로운 속성`LIST`이며 `nestedType`는 구성 요소 유형에 `RELATIONSHIP` 있습니다. 계층 구조는 이름 앞에 다음 중 하나가 붙은 속성에 매핑됩니다.
  - 사용자 지정 워크스페이스 - *`Hierarchy_hierarchyId`*
  - 기본 워크스페이스 - *`hierarchyId`*

자산

AWS IoT TwinMaker 는의 각 자산에 대해 새 개체를 생성합니다 AWS IoT SiteWise.

- `entityId`는 `assetId`의와 동일합니다 AWS IoT SiteWise.
- 이러한 개체에는 `sitewiseBase`(이)라는 단일 구성 요소가 있으며, 이 구성 요소에는 이 자산의 자산 모델에 해당하는 구성 요소 유형이 있습니다.
- 속성 별칭 또는 측정 단위 설정과 같은 모든 자산 수준 무시도 AWS IoT TwinMaker의 개체에 반영됩니다.

기본 워크스페이스만

다음 자산은 동기화되어 기본 워크스페이스인 에서만 사용할 수 있습니다IoTSiteWiseDefaultWorkspace.

AssetModelComponents

AWS IoT TwinMaker 는의 각 AssetModelComponents에 대해 새 구성 요소 유형을 생성합니다 AWS IoT SiteWise.

- TypeId 자산 모델의 구성 요소는 패턴을 사용합니다assetModelId.
- 자산 모델의 각 속성은 구성 요소 유형의 새 속성이며 속성 이름은 propertyId와(과) 같습니다. 의 속성 이름은 속성 정의displayName에 로 AWS IoT SiteWise 저장됩니다.
- 자산 모델의 각 계층 구조는 유형의 새로운 속성LIST이며 nestedType는 구성 요소 유형에 RELATIONSHIP 있습니다. 계층 구조는 hierarchyId(이)라는 이름 앞에 붙은 속성에 매핑됩니다.

AssetModelCompositeModel

AWS IoT TwinMaker 는의 각 AssetModelCompositeModel에 대해 새 구성 요소 유형을 생성합니다 AWS IoT SiteWise.

- TypeId 자산 모델의 구성 요소는 패턴을 사용합니다assetModelId_assetModelCompositeModelId.
- 자산 모델의 각 속성은 구성 요소 유형의 새 속성이며 속성 이름은 propertyId와(과) 같습니다. 의 속성 이름은 속성 정의displayName에 로 AWS IoT SiteWise 저장됩니다.

AssetCompositeModels

AWS IoT TwinMaker 는의 각 AssetCompositeModel에 대해 새 복합 구성 요소를 생성합니다 AWS IoT SiteWise.

- componentName는 assetModelCompositeModelId의와 동일합니다 AWS IoT SiteWise.

리소스가 동기화되지 않음

다음 리소스는 동기화되지 않습니다.

동기화되지 않은 자산 및 자산 모델

- 경보 모델은 compositeModels로 동기화되지만 경보와 관련된 자산의 해당 데이터는 동기화되지 않습니다.

- AWS IoT SiteWise 데이터 스트림은 동기화되지 않습니다. 자산 모델에서 모델링된 속성만 동기화됩니다.
- 속성, 측정, 변환, 집계 및 메타데이터 계산(예: 공식 및 창) 의 속성 값은 동기화되지 않습니다. 별칭, 측정 단위 및 데이터 유형과 같은 속성에 대한 메타데이터만 동기화됩니다. 일반 AWS IoT TwinMaker 데이터 커넥터 API인 GetPropertyValueHistory를 사용하여 값을 쿼리할 수 있습니다.

에서 동기화된 엔터티 및 구성 요소 유형 사용 AWS IoT TwinMaker

자산이 동기화되면 AWS IoT SiteWise동기화된 구성 요소 유형만 읽습니다 AWS IoT TwinMaker. 모든 업데이트 또는 삭제 작업은에서 수행해야 하며 AWS IoT SiteWise, 이러한 변경 사항은 syncJob이 여전히 활성 상태인 AWS IoT TwinMaker 경우에 동기화됩니다.

동기화된 개체와 AWS IoT SiteWise 기본 구성 요소도 읽기 전용입니다 AWS IoT TwinMaker. 설명과 같은 개체 수준이 없거나 `entityName`이(가) 업데이트되는 한 추가적으로 동기화되지 안흔 구성 요소를 동기화된 개체에 추가할 수 있습니다.

동기화된 개체와 상호 작용하는 방법에는 몇 가지 제한이 적용됩니다. 동기화된 개체의 계층 구조에서 동기화된 개체 아래에 하위 개체를 생성할 수 없습니다. 또한 동기화된 구성 요소 유형에서 확장되는 동기화되지 않은 구성 요소 유형을 생성할 수 없습니다.

> **Note**
>
> 자산이에서 삭제 AWS IoT SiteWise 되거나 동기화 작업을 삭제하면 개체와 함께 추가 구성 요소가 삭제됩니다.

Grafana 대시보드에서 이러한 동기화된 개체를 사용하고 일반 개체처럼 장면 컴포저에 태그로 추가할 수 있습니다. 이러한 동기화된 개체에 대해 지식 그래프 쿼리를 발행할 수도 있습니다.

> **Note**
>
> 수정하지 않고 동기화된 개체에는 요금이 부과되지 않지만, AWS IoT TwinMaker에서 개체를 변경한 경우 해당 개체에 대한 요금이 부과됩니다. 예를 들어 동기화되지 않은 구성 요소를 동기화된 개체에 추가하면 이제 해당 개체에 요금이 부과됩니다 AWS IoT TwinMaker. 자세한 내용은 AWS IoT TwinMaker 요금을 참조하십시오.

동기화 상태 및 오류 분석

이 주제에서는 동기화 오류 및 상태를 분석하는 방법에 대한 지침을 제공합니다.

> **Important**
>
> 사용자 지정 워크스페이스와 기본 워크스페이스의 차이점에 대한 the section called “사용자 지정 워크스페이스와 기본 워크스페이스의 차이점” 자세한 내용은 섹션을 참조하세요.

작업 상태 동기화

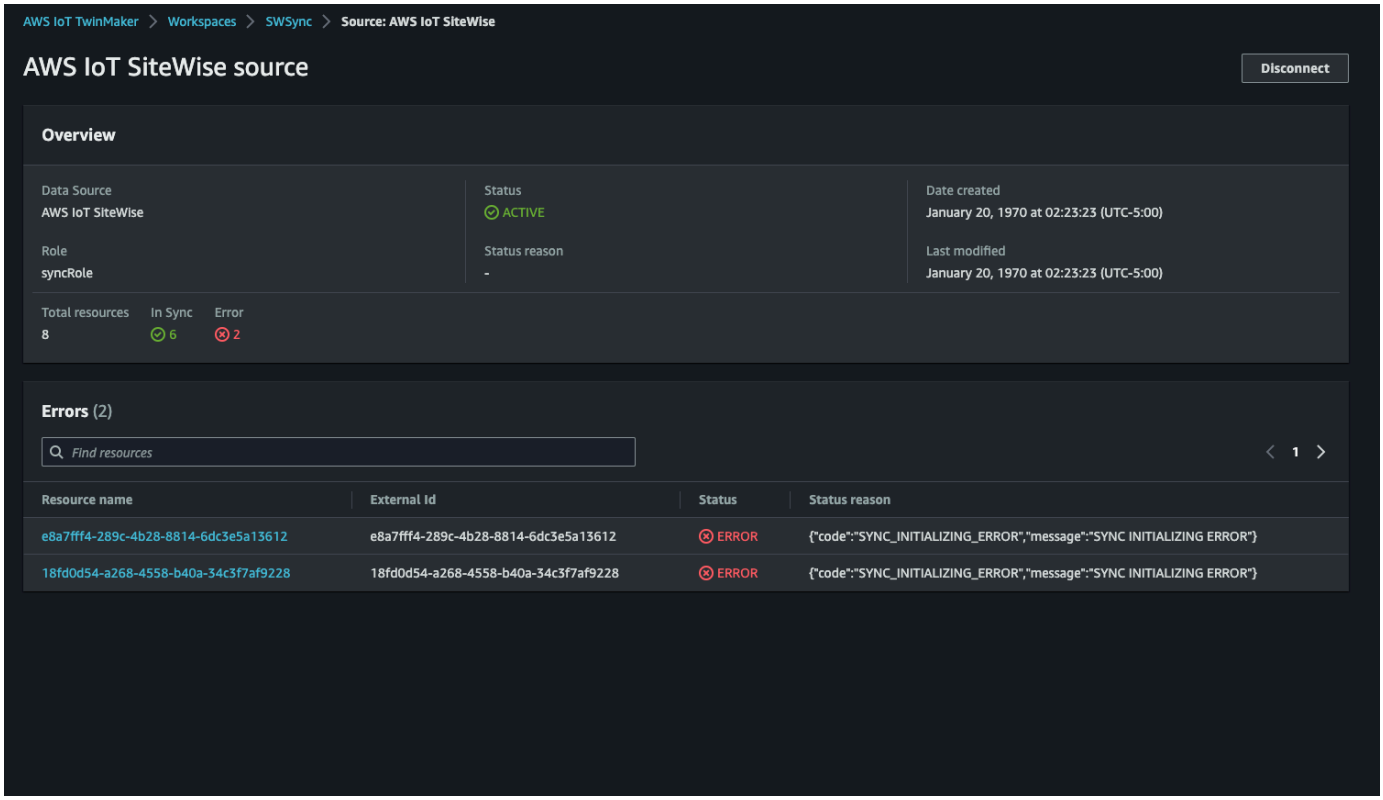
동기화 작업은 그 상태에 따라 다음 상태 중 하나를 가집니다.

- 동기화 작업 `CREATING` 상태는 작업이 권한을 확인하고 동기화를 준비 AWS IoT SiteWise 하기 위해에서 데이터를 로드하고 있음을 의미합니다.
- 동기화 작업 `INITIALIZING` 상태는의 모든 기존 리소스 AWS IoT SiteWise 가 동기화됨을 의미합니다 AWS IoT TwinMaker. 사용자가 많은 수의 자산과 자산 모델을 AWS IoT SiteWise에 보유하고 있는 경우, 이 단계를 완료하는 데 시간이 더 오래 걸릴 수 있습니다. AWS IoT TwinMaker 콘솔에서 동기화 작업을 확인하거나 `ListSyncResources` API를 직접적으로 호출하여 동기화된 리소스 수를 모니터링할 수 있습니다.
- 동기화 작업 `ACTIVE` 상태는 초기화 단계가 완료되었음을 의미합니다. 이제 작업을 통해 AWS IoT SiteWise에서 모든 새 업데이트를 동기화할 준비가 되었습니다.
- 동기화 작업 `ERROR` 상태는 이전 상태 중 하나와 관련된 오류를 나타냅니다. 오류 메시지를 검토합니다. IAM 역할 설정에 문제가 있을 수 있습니다. 새 IAM 역할을 사용하려면 오류가 발생한 동기화 작업을 삭제하고 새 역할로 새 동기화 작업을 생성합니다.

동기화 오류는 작업 영역의 Entity 모델 소스 테이블에서 액세스할 수 있는 모델 소스 페이지에 나타납니다. 모델 소스 페이지에는 동기화에 실패한 리소스 목록이 표시됩니다. 대부분의 오류는 동기화 작업에 의해 자동으로 재시도되지만, 리소스에 작업이 필요한 경우 해당 `ERROR` 상태가 유지됩니다. ListSyncResources API를 사용하여 오류 목록을 가져올 수도 있습니다.

현재 원본에 대해 나열된 오류를 모두 보려면 다음 절차를 따릅니다.

1. AWS IoT TwinMaker 콘솔의 작업 영역으로 이동합니다.
2. 개체 모델 AWS IoT SiteWise 소스 모달에 나열된 소스를 선택하여 자산 동기화 세부 정보 페이지를 엽니다.



3. 이전 스크린샷에서 볼 수 있듯이 오류가 지속되는 모든 리소스는 오류 테이블에 나열됩니다. 이 표를 사용하여 특정 리소스와 관련된 오류를 추적하고 수정할 수 있습니다.

다음과 같은 오류가 있습니다.

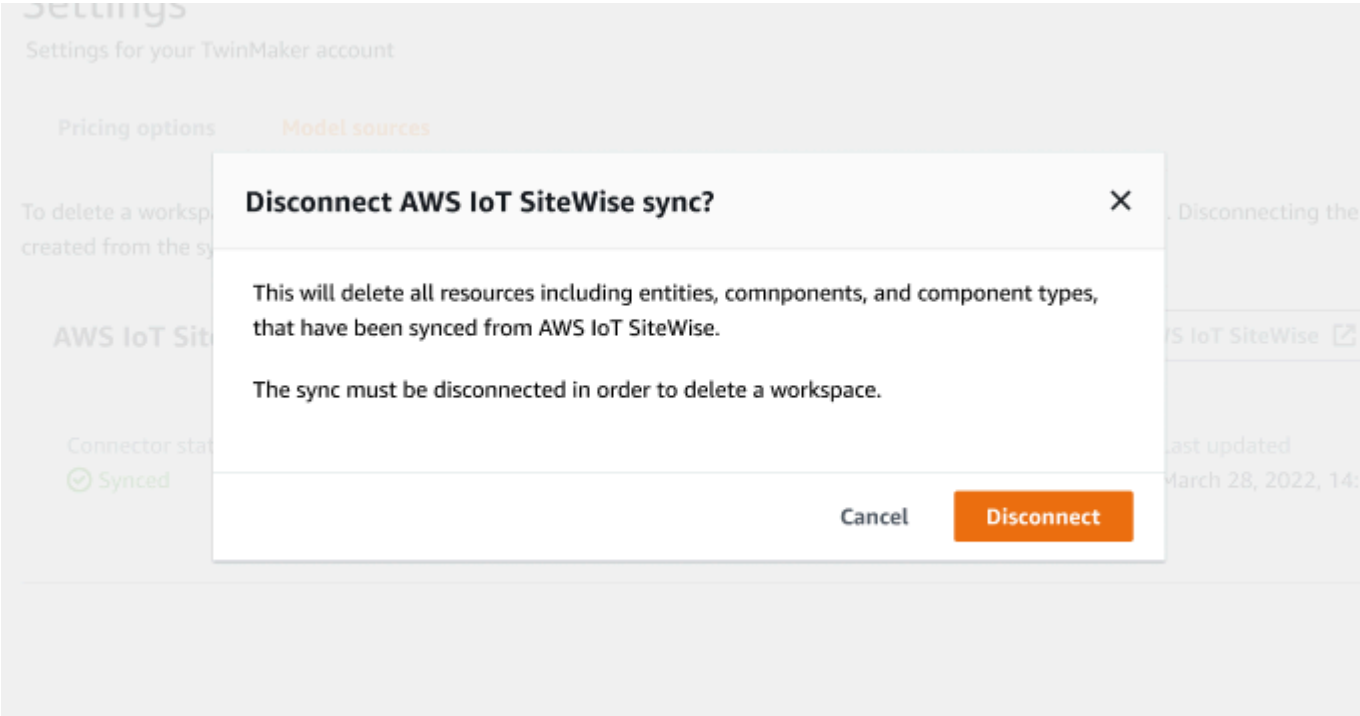
- AWS IoT SiteWise 는 중복 자산 이름을 지원하지만,는 동일한 상위 개체가 아닌 ROOT 수준에서 AWS IoT TwinMaker 만 자산 이름을 지원합니다. 상위 엔터티 아래에 동일한 이름의 자산이 두 개 있는 경우 그 중 AWS IoT SiteWise하나가 동기화되지 않습니다. 이 오류를 해결하려면 동기화 AWS IoT SiteWise 하기 전에에서 자산 중 하나를 삭제하거나 다른 상위 자산으로 이동합니다.
- 자산 ID와 ID가 동일한 엔터티가 AWS IoT SiteWise 이미 있는 경우 기존 엔터티를 삭제할 때까지 해당 자산이 동기화되지 않습니다.

동기화 작업 삭제

동기화 작업을 삭제하려면 다음 절차에 따르십시오.

> **Important**
>
> 사용자 지정 워크스페이스와 기본 워크스페이스의 차이점에 대한 the section called “사용자 지정 워크스페이스와 기본 워크스페이스의 차이점” 자세한 내용은 섹션을 참조하세요.

1. AWS IoT TwinMaker 콘솔로 이동합니다.
2. 동기화 작업을 삭제할 곳에서 작업공간을 엽니다.
3. 개체 모델 소스에서 AWS IoT SiteWise 소스를 선택하여 소스 세부 정보 페이지를 엽니다.
4. 동기화 작업을 중지하려면 연결 끊기를 선택합니다. 동기화 작업을 완전히 삭제하는 것을 선택했는지 확인하십시오.



일단 동기화 작업이 삭제되면 같은 작업공간이나 다른 작업공간에서 동기화 작업을 다시 생성할 수 있습니다.

해당 워크스페이스에 동기화 작업이 있는 경우 워크스페이스를 삭제할 수 없습니다. 작업 영역을 삭제하기 전에 먼저 동기화 작업을 삭제하십시오.

동기화 작업을 삭제하는 동안 오류가 발생하는 경우, 동기화 작업은 해당 `DELETING` 상태로 유지되며 자동으로 재시도됩니다. 이제 리소스 삭제와 관련된 오류가 발생하는 경우, 동기화된 개체 또는 구성 요소 유형을 수동으로 삭제할 수 있습니다.

> **Note**
>
> 에서 동기화된 모든 리소스 AWS IoT SiteWise 가 먼저 삭제된 다음 동기화 작업 자체가 삭제됩니다.

자산 동기화 제한

> **Important**
>
> 사용자 지정 워크스페이스와 기본 워크스페이스의 차이점에 대한 the section called “사용자 지정 워크스페이스와 기본 워크스페이스의 차이점” 자세한 내용은 섹션을 참조하세요.

AWS IoT SiteWise 할당량이 기본 AWS IoT TwinMaker 할당량보다 높기 때문에 AWS IoT SiteWise에서 동기화되는 항목 및 구성 요소 유형에 대한 다음 한도를 늘리고 있습니다.

- 워크스페이스에는 1,000개의 자산 모델만 동기화할 수 있으므로 1,000개의 동기화된 구성 요소 유형이 있습니다 AWS IoT SiteWise.
- 워크스페이스에는 100,000개의 자산만 동기화할 수 있으므로 100,000개의 동기화된 엔터티가 있습니다 AWS IoT SiteWise.
- 상위 개체당 최대 하위 개체 2000개 단일 상위 자산당 2000개의 하위 자산을 동기화합니다.

  > **Note**
  >
  > GetEntity API는 계층 구조 속성에 대해 처음 50개의 자식 개체만 반환하지만 GetPropertyValue API를 사용하여 모든 하위 개체의 목록을 페이지 매김하고 검색할 수 있습니다.

- 자산 모델을 총 600개의 속성 및 계층 구조와 동기화할 수 AWS IoT SiteWise있는 동기화된 구성 요소당 600개의 속성.

> **Note**
>
> 이러한 제한은 동기화된 개체에만 적용할 수 있습니다. 동기화되지 않은 리소스에 대해 이러한 한도를 늘려야 하는 경우 할당량 증가를 요청하십시오.

AWS IoT TwinMaker Grafana 대시보드 통합

AWS IoT TwinMaker 는 애플리케이션 플러그인을 통한 Grafana 통합을 지원합니다. Grafana 버전 10.4.0 이상 버전을 사용하여 디지털 트윈 애플리케이션과 상호 작용합니다. AWS IoT TwinMaker 플러그인은 디지털 트윈 데이터에 연결하기 위한 사용자 지정 패널, 대시보드 템플릿 및 데이터 소스를 제공합니다.

Grafana 온보딩 및 대시보드 권한 설정 방법에 관한 자세한 내용은 다음 주제를 참조하십시오.

주제

- Grafana 장면 뷰어를 위한 CORS 구성
- Grafana 환경 설정
- 대시보드 IAM 역할 생성
- AWS IoT TwinMaker 비디오 플레이어 정책 생성

> **Note**
>
> Grafana 사용자 인터페이스가 버킷에서 리소스를 로드할 수 있도록 Amazon S3 버킷의 CORS(원본 간 리소스 공유) 구성을 수정해야 합니다. 해당 지침은 Grafana 장면 뷰어를 위한 CORS 구성을(를) 참조하십시오.

AWS IoT TwinMaker Grafana 플러그인에 대한 자세한 내용은 AWS IoT TwinMaker 앱 설명서를 참조하세요.

Grafana 플러그인의 주요 구성 요소에 대한 자세한 내용은 다음을 참조하십시오.

- AWS IoT TwinMaker 데이터 원본
- 대시보드 템플릿
- 장면 뷰어 패널
- 동영상 플레이어 패널

Grafana 장면 뷰어를 위한 CORS 구성

AWS IoT TwinMaker Grafana 플러그인에는 CORS(크로스 오리진 리소스 공유) 구성이 필요합니다. 이를 통해 Grafana 사용자 인터페이스가 Amazon S3 버킷에서 리소스를 로드할 수 있습니다. CORS 구성이 없으면 Grafana 도메인이 Amazon S3 버킷의 리소스에 액세스할 수 없으므로 장면 뷰어에 “네트워크 장애로 인한 3D 장면 로드 실패”라는 오류 메시지가 표시됩니다.

CORS로 Amazon S3 버킷을 구성하려면 다음 단계를 따르십시오.

1. IAM 콘솔에 로그인한 다음 Amazon S3 콘솔을 엽니다.
2. 버킷 목록에서 AWS IoT TwinMaker 워크스페이스의 리소스 버킷으로 사용하는 버킷의 이름을 선택합니다.
3. 권한을 선택합니다.
4. CORS(원본 간 리소스 공유) 섹션에서 편집을 선택합니다.
5. CORS 구성 편집기 텍스트 상자에서 Grafana 작업 영역 도메인 *GRAFANA-WORKSPACE-DOMAIN*을(를) 사용자 도메인으로 대체하여 다음 JSON CORS 구성을 입력하거나 복사하여 붙여넣습니다.

> **Note**
>
> 별표 * 문자는 "AllowedOrigins": JSON 요소의 시작 부분에 유지해야 합니다.

```
    [
{
    "AllowedHeaders": [
        "*"
    ],
    "AllowedMethods": [
        "GET",
        "PUT",
        "POST",
        "DELETE",
        "HEAD"
    ],
    "AllowedOrigins": [
        "*GRAFANA-WORKSPACE-DOMAIN"
    ],
```

```
        "ExposeHeaders": [
            "ETag"
        ]
    }
]
```

6. 변경 사항 저장을 선택하여 CORS 구성을 완료합니다.

Amazon S3 버킷을 사용하는 CORS에 대한 자세한 내용은 CORS(Cross-Origin Resource Sharing) 사용을 참조하세요.

Grafana 환경 설정

Amazon Managed Grafana를 완전 관리형 서비스에 사용하거나, 직접 관리하는 Grafana 환경을 설정할 수 있습니다. Amazon Managed Grafana를 사용하면 필요에 따라 오픈 소스 Grafana를 신속하게 배포, 운영 및 확장할 수 있습니다. 또는 자체 인프라를 설정하여 Grafana 서버를 관리할 수 있습니다.

Grafana 환경 옵션에 대한 자세한 내용은 다음 주제를 참조하십시오.

- Amazon Managed Grafana
- 자체 관리형 Grafana

Amazon Managed Grafana

Amazon Managed Grafana는 AWS IoT TwinMaker 플러그인을 제공하므로 Grafana AWS IoT TwinMaker 와 빠르게 통합할 수 있습니다. Amazon Managed Grafana는 사용자를 대신하여 Grafana 서버를 관리하므로 사용자는 하드웨어 또는 기타 Grafana 인프라를 구축, 패키징 또는 배포하지 않고도 데이터를 시각화할 수 있습니다. Amazon Managed Grafana에 대한 자세한 내용은 Amazon Managed Grafana란 무엇입니까?를 참조하십시오.

> **Note**
>
> Amazon Managed Grafana는 현재 AWS IoT TwinMaker Grafana 플러그인 버전 1.3.1을 지원합니다.

Amazon Managed Grafana 사전 요구 사항

Amazon Managed Grafana 대시보드 AWS IoT TwinMaker 에서를 사용하려면 먼저 다음 사전 조건을 완료하세요.

- AWS IoT TwinMaker 워크스페이스를 생성합니다. 워크스페이스 생성에 대한 자세한 내용은 시작하기를 AWS IoT TwinMaker 참조하세요.

> **Note**
>
> AWS 관리 콘솔에서 Amazon Managed Grafana 워크스페이스를 처음 생성하면 AWS IoT TwinMaker 이 나열되지 않습니다. 하지만 해당 플러그인은 이미 모든 작업 영역에 설치되어 있습니다. 오픈 소스 Grafana 플러그인 목록에서 AWS IoT TwinMaker 플러그인을 찾을 수 있습니다. 데이터소스 페이지에서 데이터소스 추가를 선택하여 AWS IoT TwinMaker 데이터소스를 찾을 수 있습니다.

Amazon Managed Grafana 작업 영역을 생성하면 Grafana 인스턴스에 대한 권한을 관리하기 위해 IAM 역할이 자동으로 생성됩니다. 이를 작업 영역 IAM 역할이라고 합니다. Grafana의 모든 AWS IoT TwinMaker 데이터소스를 구성하는 데 사용할 인증 공급자 옵션입니다. Amazon Managed Grafana는 AWS IoT TwinMaker에 대한 권한 자동 추가를 지원하지 않으므로 이러한 권한을 수동으로 설정해야 합니다. 수동 권한 설정에 대한 자세한 내용은 대시보드 IAM 역할 생성를 참조하십시오.

자체 관리형 Grafana

Grafana를 실행하기 위해 자체 인프라를 호스팅하도록 선택할 수 있습니다. 컴퓨터에서 Grafana를 로컬로 실행하는 방법에 대한 자세한 내용은 Grafana 설치를 참조하십시오. AWS IoT TwinMaker 플러그인은 공개 Grafana 카탈로그에서 사용할 수 있습니다. Grafana 환경에 이 플러그인을 설치하는 방법에 대한 자세한 내용은 AWS IoT TwinMaker 앱을 참조하십시오.

Grafana를 로컬에서 실행하면 대시보드를 쉽게 공유하거나 여러 사용자에게 액세스를 제공할 수 없습니다. 로컬 Grafana를 사용하여 대시보드를 공유하는 방법에 대한 스크립트로 작성된 빠른 시작 가이드는 AWS IoT TwinMaker 샘플 리포지토리를 참조하십시오. 이 리소스는 Cloud9에서 Grafana 환경을 호스팅하고 퍼블릭 엔드포인트에서 Amazon EC2를 호스팅하는 과정을 안내합니다.

TwinMaker 데이터 소스를 구성하는 데 사용할 인증 공급자를 결정해야 합니다. 기본 자격 증명 체인을 기반으로 환경의 자격 증명을 구성합니다(기본 자격 증명 공급자 체인 사용 참조). 기본 자격 증명은 모든 사용자 또는 역할의 영구 자격 증명일 수 있습니다. 예를 들어 Amazon EC2에서 Grafana를 실행하

는 경우 기본 자격 증명 체인은 인증 공급자가 되는 Amazon EC2 실행 역할에 액세스할 수 있습니다. 대시보드 IAM 역할 생성 단계에서는 인증 제공자의 IAM Amazon 리소스 이름(ARN)이 필요합니다.

대시보드 IAM 역할 생성

를 사용하면 Grafana 대시보드에서 데이터 액세스를 제어할 AWS IoT TwinMaker수 있습니다. Grafana 대시보드 사용자는 데이터를 보고, 경우에 따라 데이터를 쓰기 위해 다양한 권한 범위를 가져야 합니다. 예를 들어 알람 운영자에게는 동영상 보기 권한이 없는 반면 관리자는 모든 리소스에 대한 권한이 있을 수 있습니다. Grafana는 자격 증명과 IAM 역할이 제공되는 데이터 소스를 통해 권한을 정의합니다. AWS IoT TwinMaker 데이터 소스는 해당 역할에 대한 권한이 있는 AWS 자격 증명을 가져옵니다. IAM 역할이 제공되지 않은 경우 Grafana는 자격 증명의 범위를 사용하며, 이는 축소할 수 없습니다 AWS IoT TwinMaker.

Grafana에서 AWS IoT TwinMaker 대시보드를 사용하려면 IAM 역할을 생성하고 정책을 연결합니다. 다음 템플릿을 사용하여 이러한 정책을 생성할 수 있습니다.

IAM 정책 생성

IAM 콘솔에서 *YourWorkspaceId*DashboardPolicy이라는 IAM 정책을 생성합니다. 이 정책은 작업 영역에 Amazon S3 버킷 및 AWS IoT TwinMaker 리소스에 대한 액세스 권한을 부여합니다. 구성 요소에 AWS IoT Greengrass 대해 구성된 Kinesis Video Streams 및 자산에 대한 권한이 필요한 Amazon Kinesis Video Streams용 Edge Connector를 사용하기로 결정할 수도 있습니다. AWS IoT SiteWise 사용 사례에 맞게 다음 정책 템플릿 중 하나를 선택합니다.

1: 동영상 권한 정책 없음

Grafana Video Player 패널을 사용하지 않으려면 다음 템플릿을 사용하여 정책을 생성하십시오.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/*",
                "arn:aws:s3:::amzn-s3-demo-bucket"
```

```
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iottwinmaker:Get*",
                "iottwinmaker:List*"
            ],
            "Resource": [
                "arn:aws:iottwinmaker:us-
east-1:111122223333:workspace/workspaceId",
                "arn:aws:iottwinmaker:us-
east-1:111122223333:workspace/workspaceId/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "iottwinmaker:ListWorkspaces",
            "Resource": "*"
        }
    ]
}
```

Amazon S3 버킷이 각 작업 영역에 대해 생성됩니다. 여기에는 대시보드에서 볼 수 있는 3D 모델 및 장면이 포함되어 있습니다. SceneViewer 패널은이 버킷에서 항목을 로드합니다.

2. 동영상 권한 정책의 범위 축소

   Grafana의 Video Player 패널에서 액세스를 제한하려면 Amazon Kinesis Video Streams용 AWS IoT Greengrass Edge Connector 리소스를 태그별로 그룹화합니다. 동영상 리소스 권한 범위 축소에 대한 자세한 내용은 AWS IoT TwinMaker 비디오 플레이어 정책 생성을(를) 참조하십시오.

3. 모든 동영상 권한

   동영상을 그룹화하고 싶지 않은 경우 Grafana Video Player에서 모든 동영상에 액세스할 수 있도록 설정할 수 있습니다. Grafana 워크스페이스에 액세스할 수 있는 모든 사용자는 계정의 모든 스트림에 대해 비디오를 재생할 수 있으며 모든 AWS IoT SiteWise 자산에 대한 읽기 전용 액세스 권한을 가질 수 있습니다. 여기에는 향후 생성되는 모든 리소스가 포함됩니다.

   다음 템플릿을 사용하여 정책을 생성합니다.

   JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::bucketName/*",
                "arn:aws:s3:::bucketName"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iottwinmaker:Get*",
                "iottwinmaker:List*"
            ],
            "Resource": [
                "arn:aws:iottwinmaker:us-
east-1:111122223333:workspace/workspaceId",
                "arn:aws:iottwinmaker:us-
east-1:111122223333:workspace/workspaceId/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "iottwinmaker:ListWorkspaces",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kinesisvideo:GetDataEndpoint",
                "kinesisvideo:GetHLSStreamingSessionURL"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:GetAssetPropertyValue",
                "iotsitewise:GetInterpolatedAssetPropertyValues"
            ],
```

```
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:BatchPutAssetPropertyValue"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "aws:ResourceTag/EdgeConnectorForKVS": "*workspaceId*"
                }
            }
        }
    ]
}
```

이 샘플 정책은 다음 권한을 제공합니다.

- 장면을 로드하기 위한 S3 버킷에 대한 읽기 전용 액세스.
- 워크스페이스의 모든 엔터티 및 구성 요소에 AWS IoT TwinMaker 대한에 대한 읽기 전용 액세스 권한입니다.
- 계정의 모든 Kinesis Video Streams 동영상을 스트리밍할 수 있는 읽기 전용 액세스.
- 계정에 있는 모든 AWS IoT SiteWise 자산의 속성 값 기록에 대한 읽기 전용 액세스 권한입니다.
- 키EdgeConnectorForKVS와 값으로 태그가 지정된 AWS IoT SiteWise 자산의 속성으로 데이터를 수집합니다workspaceId.

엣지에서 카메라 AWS IoT SiteWise 자산 요청 비디오 업로드에 태그 지정

Grafana의 동영상 플레이어를 사용하면 사용자는 엣지 캐시에서 Kinesis Video Streams로 동영상을 업로드하도록 수동으로 요청할 수 있습니다. Amazon Kinesis Video Streams용 AWS IoT Greengrass Edge Connector와 연결되어 있고 키로 태그가 지정된 모든 AWS IoT SiteWise 자산에 대해이 기능을 켤 수 있습니다EdgeConnectorForKVS.

태그 값은 다음 `.` `:` `+` `=` `@` `_` `/` `-` 문자로 구분된 WorkspaceID 목록일 수 있습니다. 예를 들어 워크 AWS IoT TwinMaker 스페이스 전체에서 Amazon Kinesis Video Streams용 AWS IoT Greengrass 엣지 커넥터와 연결된 AWS IoT SiteWise 자산을 사용하려면 다음 패턴을 따르는 태그를 사용할 수 있습니다WorkspaceA/WorkspaceB/WorkspaceC. Grafana 플러그인은 the AWS IoT TwinMaker workspaceId가 AWS IoT SiteWise 자산 데이터 수집을 그룹화하는 데 사용되도록 강제합니다.

대시보드 정책에 더 많은 권한을 추가

AWS IoT TwinMaker Grafana 플러그인은 인증 공급자를 사용하여 생성한 대시보드 역할에서 AssumeRole을 호출합니다. 내부적으로 플러그인은 AssumeRole 호출에서 세션 정책을 사용하여 액세스할 수 있는 가장 높은 권한 범위를 제한합니다. 세션 정책에 대한 자세한 정보는 세션 정책을 참조하십시오.

이는 AWS IoT TwinMaker 작업 영역의 대시보드 역할에 적용할 수 있는 최대 허용 정책입니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::bucketName/*",
                "arn:aws:s3:::bucketName"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iottwinmaker:Get*",
                "iottwinmaker:List*"
            ],
            "Resource": [
                "arn:aws:iottwinmaker:us-
east-1:111122223333:workspace/workspaceId",
                "arn:aws:iottwinmaker:us-
east-1:111122223333:workspace/workspaceId/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "iottwinmaker:ListWorkspaces",
            "Resource": "*"
        },
```

```
        {
            "Effect": "Allow",
            "Action": [
                "kinesisvideo:GetDataEndpoint",
                "kinesisvideo:GetHLSStreamingSessionURL"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:GetAssetPropertyValue",
                "iotsitewise:GetInterpolatedAssetPropertyValues"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:BatchPutAssetPropertyValue"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "aws:ResourceTag/EdgeConnectorForKVS": "*workspaceId*"
                }
            }
        }
    ]
}
```

더 Allow 많은 권한을 가진 문을 추가하면 AWS IoT TwinMaker 플러그인에서 작동하지 않습니다. 이는 플러그인이 필요한 최소한의 권한을 사용하도록 하기 위한 것입니다.

하지만 권한 범위를 더 좁힐 수 있습니다. 자세한 내용은 AWS IoT TwinMaker 비디오 플레이어 정책 생성을(를) 참조하십시오.

Grafana 대시보드 IAM 역할 생성

IAM 콘솔에서 *YourWorkspaceId*DashboardRole이라는 IAM 역할을 생성합니다. 역할에 *YourWorkspaceId*DashboardPolicy을(를) 연결하십시오.

대시보드 역할의 신뢰 정책을 편집하려면 Grafana 인증 공급자가 대시보드 역할에 대해 `AssumeRole`을 직접적으로 호출할 수 있는 권한을 부여해야 합니다. 다음 템플릿으로 신뢰 정책을 업데이트하십시오.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "ARN of Grafana authentication provider"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Grafana 환경 생성 및 인증 제공자 찾기에 대한 자세한 내용은 Grafana 환경 설정을(를) 참조하십시오.

AWS IoT TwinMaker 비디오 플레이어 정책 생성

다음은 Grafana의 AWS IoT TwinMaker 플러그인에 필요한 모든 동영상 권한이 포함된 정책 템플릿입니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/*",
                "arn:aws:s3:::amzn-s3-demo-bucket"
```

```
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iottwinmaker:Get*",
                "iottwinmaker:List*"
            ],
            "Resource": [
                "arn:aws:iottwinmaker:us-
east-1:111122223333:workspace/workspaceId",
                "arn:aws:iottwinmaker:us-
east-1:111122223333:workspace/workspaceId/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "iottwinmaker:ListWorkspaces",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kinesisvideo:GetDataEndpoint",
                "kinesisvideo:GetHLSStreamingSessionURL"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:GetAssetPropertyValue",
                "iotsitewise:GetInterpolatedAssetPropertyValues"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:BatchPutAssetPropertyValue"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
```

```
                    "aws:ResourceTag/EdgeConnectorForKVS": "*workspaceId*"
                }
            }
        }
    ]
}
```

전체 정책에 대한 자세한 내용은 IAM 정책 생성 항목의 모든 동영상 권한 정책 템플릿을 참조하십시오.

리소스에 대한 액세스 범위 좁히기

Grafana의 동영상 플레이어 패널은 Kinesis Video Streams 및 IoT SiteWise를 직접 호출하여 완벽한 동영상 재생 환경을 제공합니다. AWS IoT TwinMaker 워크스페이스와 연결되지 않은 리소스에 대한 무단 액세스를 방지하려면 워크스페이스 대시보드 역할에 대한 IAM 정책에 조건을 추가합니다.

GET 권한 범위 좁히기

리소스에 태그를 지정하여 Amazon Kinesis Video Streams 및 AWS IoT SiteWise 자산의 액세스 범위를 좁힐 수 있습니다. 비디오 업로드 요청 기능을 활성화하기 위해 workspaceId를 AWS IoT TwinMaker 기반으로 AWS IoT SiteWise 카메라 자산에 이미 태그를 지정했을 수 있습니다. 엣지 주제에서 비디오 업로드를 참조하세요. 동일한 태그 키-값 페어를 사용하여 AWS IoT SiteWise 자산에 대한 GET 액세스를 제한하고 Kinesis Video Streams에 동일한 방식으로 태그를 지정할 수도 있습니다.

그런 다음 *YourWorkspaceId*DashboardPolicy의 kinesisvideo 및 Iotsitewise 명령문에 이 조건을 추가할 수 있습니다.

```
"Condition": {
  "StringLike": {
    "aws:ResourceTag/EdgeConnectorForKVS": "*workspaceId*"
  }
}
```

실제 사용 사례: 카메라 그룹화

이 시나리오에서는 공장에서 쿠키를 굽는 과정을 모니터링하는 다양한 카메라가 있습니다. 쿠키 반죽은 반죽실에서 만들고, 반죽은 냉동실에서 얼리고, 쿠키는 베이킹실에서 구워냅니다. 각 방에는 카메라가 설치되어 있으며 서로 다른 작업자 팀이 각 프로세스를 개별적으로 모니터링합니다. 각 운영자 그룹에 해당 룸에 대한 권한을 부여하고자 합니다. 쿠키 공장을 위한 디지털 트윈을 구축할 때는 단일 워크스페이스를 사용하지만 카메라 사용 권한은 공간별로 범위를 지정해야 합니다.

groupingId를 기반으로 카메라 그룹에 태그를 지정하여 이러한 권한 분리를 달성할 수 있습니다. 이 시나리오에서 groupingId는 BatterRoom(반죽실), FreezerRoom(냉동실), BakingRoom(베이킹실)입니다. 각 방의 카메라는 Kinesis Video Streams에 연결되어 있으며 키=EdgeConnectorForKVS, 값=BatterRoom 태그가 있어야 합니다. 이 값에는 다음 `. : + = @ _ / -` 문자로 구분된 그룹 목록일 수 있습니다.

*YourWorkspaceId*DashboardPolicy을(를) 수정하려면 다음 정책 설명을 사용하십시오.

```
...,
{
  "Effect": "Allow",
  "Action": [
    "kinesisvideo:GetDataEndpoint",
    "kinesisvideo:GetHLSStreamingSessionURL"
  ],
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "aws:ResourceTag/EdgeConnectorForKVS": "*groupingId*"
    }
  }
},
{
  "Effect": "Allow",
  "Action": [
    "iotsitewise:GetAssetPropertyValue",
    "iotsitewise:GetInterpolatedAssetPropertyValues"
  ],
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "aws:ResourceTag/EdgeConnectorForKVS": "*groupingId*"
    }
  }
},
...
```

이러한 문은 스트리밍 비디오 재생 및 AWS IoT SiteWise 속성 기록 액세스를 그룹의 특정 리소스로 제한합니다. 사용 사례에 따라 *groupingId*이(가) 정의됩니다. 이 시나리오에서는 roomId가 됩니다.

Scope down AWS IoT SiteWise BatchPutAssetPropertyValue 권한

이 권한을 제공하면 동영상 플레이어에서 동영상 업로드 요청 기능이 활성화됩니다. 동영상을 업로드할 때, 시간 범위를 지정하고 Grafana 대시보드 패널에서 제출을 선택하여 요청을 제출할 수 있습니다.

iotsitewise:BatchPutAssetPropertyValue 권한을 부여하려면 기본 정책을 사용하십시오.

```
...,
{
  "Effect": "Allow",
  "Action": [
    "iotsitewise:BatchPutAssetPropertyValue"
  ],
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "aws:ResourceTag/EdgeConnectorForKVS": "*workspaceId*"
    }
  }
},
...
```

사용자는이 정책을 사용하여 AWS IoT SiteWise 카메라 자산의 모든 속성에 대해 BatchPutAssetPropertyValue를 호출할 수 있습니다. 명령문의 조건에 특정 propertyId를 지정하여 권한 부여를 제한할 수 있습니다.

```
{
  ...
  "Condition": {
    "StringEquals": {
      "iotsitewise:propertyId": "propertyId"
    }
  }
  ...
}
```

Grafana의 동영상 플레이어 패널은 데이터를 VideoUploadRequest라는 측정 속성으로 수집하여 엣지 캐시에서 Kinesis Video Streams로 동영상 업로드를 시작합니다. AWS IoT SiteWise 콘솔에서이 속성의 propertyId를 찾습니다. *YourWorkspaceId*DashboardPolicy을(를) 수정하려면 다음 정책 설명을 사용하십시오.

```
...,
{
  "Effect": "Allow",
  "Action": [
    "iotsitewise:BatchPutAssetPropertyValue"
  ],
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "aws:ResourceTag/EdgeConnectorForKVS": "*workspaceId*"
    },
    "StringEquals": {
      "iotsitewise:propertyId": "VideoUploadRequestPropertyId"
    }
  }
},
...
```

이 정책은 태그가 지정된 AWS IoT SiteWise 카메라 자산의 특정 속성에 대한 데이터 수집을 제한합니다. 자세한 내용은 AWS IoT SiteWise 가 IAM과 함께 작동하는 방법을 참조하십시오.

AWS IoT TwinMaker Grafana 대시보드에 AWS IoT SiteWise 경보 연결

> **Note**
>
> 이 기능은 공용 미리 보기 버전으로 출시 중이기 때문에 변경될 수도 있습니다.

AWS IoT TwinMaker 는 AWS IoT SiteWise 및 이벤트 경보를 AWS IoT TwinMaker 구성 요소로 가져올 수 있습니다. 이를 통해 AWS IoT SiteWise 데이터 마이그레이션을 위한 사용자 지정 데이터 커넥터를 구현하지 않고도 경보 상태를 쿼리하고 경보 임계값을 구성할 수 있습니다. Grafana 플러그인을 사용하면 API를 AWS IoT TwinMaker 호출 AWS IoT TwinMaker 하거나 경보와 직접 상호 작용하지 않고도 Grafana에서 경보 상태를 시각화하고 AWS IoT SiteWise 경보 임계값을 구성할 수 있습니다.

> **Note**
>
> 지원 종료 알림: 2026 AWS 년 5월 20일에에 대한 지원이 종료됩니다 AWS IoT Events. 2026 년 5월 20일 이후에는 AWS IoT Events 콘솔 또는 AWS IoT Events 리소스에 더 이상 액세스할 수 없습니다. 자세한 내용은 AWS IoT Events 지원 종료를 참조하세요.

AWS IoT SiteWise 경보 구성 사전 조건

경보를 생성하여 Grafana 대시보드에 통합하기 전에 다음 사전 요구 사항을 검토했는지 확인하십시오.

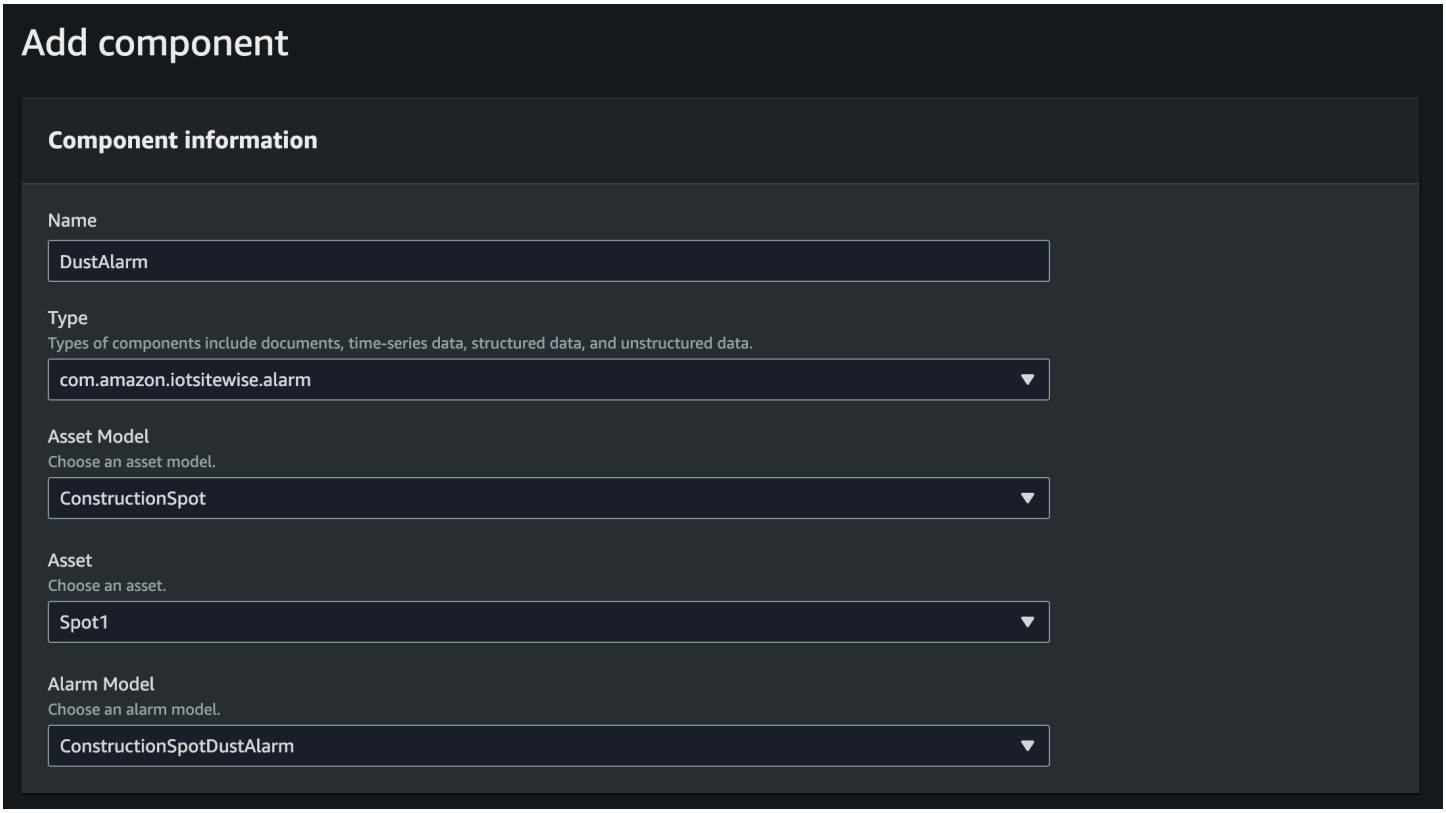
- AWS IoT SiteWise의 모델 및 자산 시스템을 숙지합니다. 자세한 내용은 AWS IoT SiteWise 사용 설명서의 자산 모델 생성 및 자산 생성을 참조하세요.
- IoT Events 경보 모델과 AWS IoT SiteWise 모델에 연결하는 방법을 숙지합니다. 자세한 내용은 AWS IoT SiteWise 사용 설명서의 AWS IoT 이벤트 경보 정의를 참조하세요.
- Grafana의 AWS IoT TwinMaker 리소스에 액세스할 수 있도록 Grafana AWS IoT TwinMaker 와 통합합니다. 자세한 내용은 단원을 참조하십시오AWS IoT TwinMaker Grafana 대시보드 통합.

AWS IoT SiteWise 경보 구성 요소 IAM 역할 정의

AWS IoT TwinMaker 는 워크스페이스 IAM 역할을 사용하여 Grafana에서 경보 임계값을 쿼리하고 구성합니다. Grafana에서 AWS IoT SiteWise 경보와 상호 작용하려면 AWS IoT TwinMaker 워크스페이스 역할에 다음 권한이 필요합니다.

```
{
"Effect": "Allow",
    "Action": [
        "iotevents:DescribeAlarmModel",
    ],
    "Resource": ["{IoTEventsAlarmModelArn}"]
},{
"Effect": "Allow",
    "Action": [
        "iotsitewise:BatchPutAssetPropertyValue"
    ],
    "Resource": ["{IoTSitewiseAssetArn}"]
}
```

AWS IoT TwinMaker 콘솔에서 AWS IoT SiteWise 자산을 나타내는 개체를 생성합니다. 를 구성 요소 유형`com.amazon.iotsitewise.alarm`으로 사용하여 해당 개체에 대한 구성 요소를 추가하고 해당 자산 및 경보 모델을 선택해야 합니다.



위의 스크린샷은 유형으로이 개체를 생성하는 예제입니다`com.amazon.iotsitewise.alarm`.

이 구성 요소를 생성하면는 AWS IoT SiteWise 및에서 관련 경보 속성을 AWS IoT TwinMaker 자동으로 가져옵니다 AWS IoT Events. 이 경보 구성 요소 유형 패턴을 반복하여 워크스페이스에 필요한 모든 자산에 대한 경보 구성 요소를 생성할 수 있습니다.

AWS IoT TwinMaker API를 통한 쿼리 및 업데이트

경보 구성 요소를 생성한 후 AWS IoT TwinMaker API를 통해 경보 상태, 임계값을 쿼리하고 경보 임계값을 업데이트할 수 있습니다.

아래는 경보 상태를 쿼리하기 위한 샘플 요청입니다.

```
aws iottwinmaker get-property-value-history --cli-input-json \
'{
    "workspaceId": "{workspaceId}",
    "entityId": "{entityId}",
    "componentName": "{componentName}",
    "selectedProperties": ["alarm_status"],
    "startTime": "{startTimeIsoString}",
```

```
    "endTime": "{endTimeIsoString}"
}'
```

아래는 경보 임계값을 쿼리하기 위한 샘플 요청입니다.

```
aws iottwinmaker get-property-value-history --cli-input-json \
'{
    "workspaceId": "{workspaceId}",
    "entityId": "{entityId}",
    "componentName": "{componentName}",
    "selectedProperties": ["alarm_threshold"],
    "startTime": "{startTimeIsoString}",
    "endTime": "{endTimeIsoString}"
}'
```

다음은 경보 임계값 업데이트를 위한 샘플 요청입니다.

```
aws iottwinmaker batch-put-property-values --cli-input-json \
'{
    "workspaceId": "{workspaceId}",
    "entries": [
        {
            "entityPropertyReference": {
                "entityId": "{entityId}",
                "componentName": "{componentName}",
                "propertyName": "alarm_threshold"
            },
            "propertyValues": [
                {
                    "value": {
                        "doubleValue": "{newThreshold}"
                    },
                    "time": "{effectiveTimeIsoString}"
                }
            ]
        }
    ]
}'
```

경보에 대한 Grafana 대시보드 구성

두 번째 쓰기 가능 대시보드 IAM 역할을 생성해야 합니다. 이는 일반적인 역할이지만 아래 예와 같이 TwinMaker 작업 영역에 작업 `iottwinmaker:BatchPutPropertyValues`을(를) 추가할 수 있는 권한이 있어야 합니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iottwinmaker:Get*",
               "iottwinmaker:List*",
                "iottwinmaker:BatchPutPropertyValues"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "iottwinmaker:ListWorkspaces",
            "Resource": "*"
        }
    ]
}
```

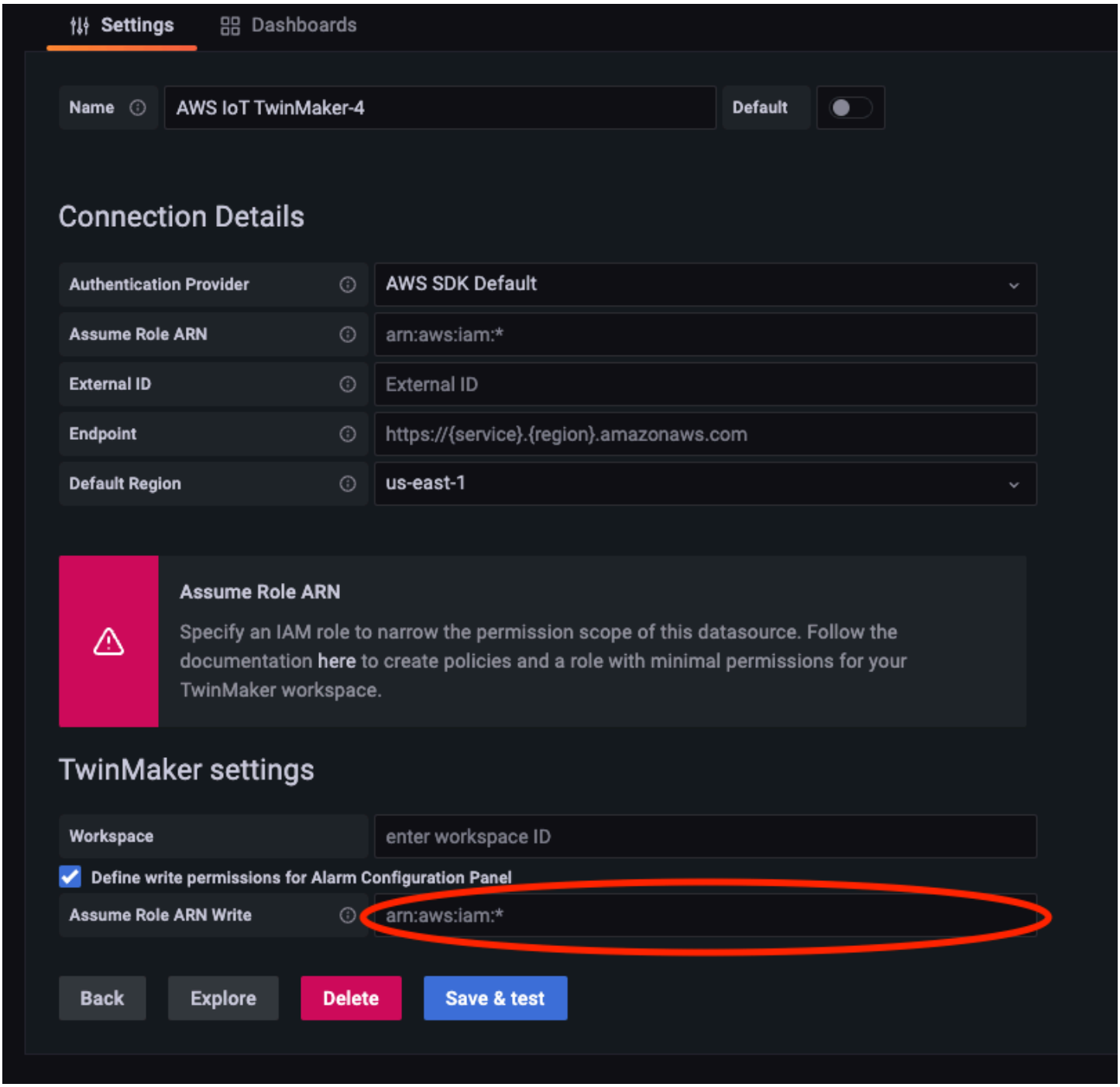
대신에 IAM 역할 끝에 이 명령문을 추가할 수도 있습니다.

```
{
    "Effect": "Allow",
    "Action": [
        "iottwinmaker:BatchPutPropertyValues"
    ],
    "Resource": [
        "{workspaceArn}",
```

```
            "{workspaceArn}/*"
        ]
}
```

데이터 소스에는 생성한 대시보드 쓰기 역할과 함께 쓰기 arn이 설정되어 있어야 합니다.

IAM 역할을 수정한 후 Grafana 대시보드에 로그인하여 업데이트된 역할 ARN을 수임합니다. 경보 구성 패널의 쓰기 권한 정의 확인란을 선택하고 쓰기 역할의 ARN에 복사합니다.



경보 시각화용 Grafana 대시보드 사용

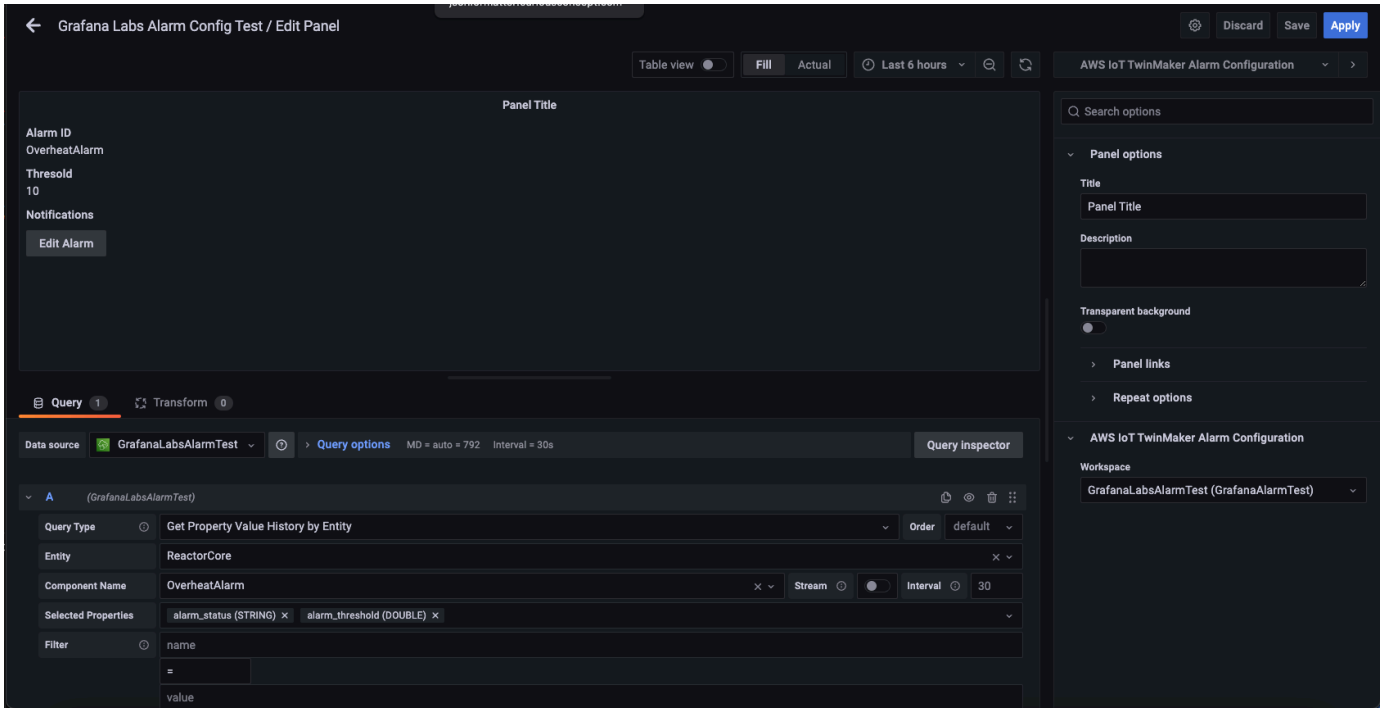
다음 절차를 사용하여 대시보드에 경보 구성 패널을 추가하고 구성하십시오.

1. 패널 옵션에서 작업 영역을 선택합니다.
2. 쿼리 구성에서 데이터소스를 설정합니다.
3. 다음 쿼리 유형: `Get Property Value History by Entity`을(를) 사용하십시오.
4. 경보를 추가할 개체 또는 개체 변수를 선택합니다.
5. 개체를 선택한 후 속성을 적용할 구성 요소 또는 구성 요소 변수를 선택합니다.
6. 속성으로: `alarm_status` 및 `alarm_threshold`을(를) 선택합니다.

   연결되면 경보 ID의 ID와 현재 임계값이 표시될 것입니다.

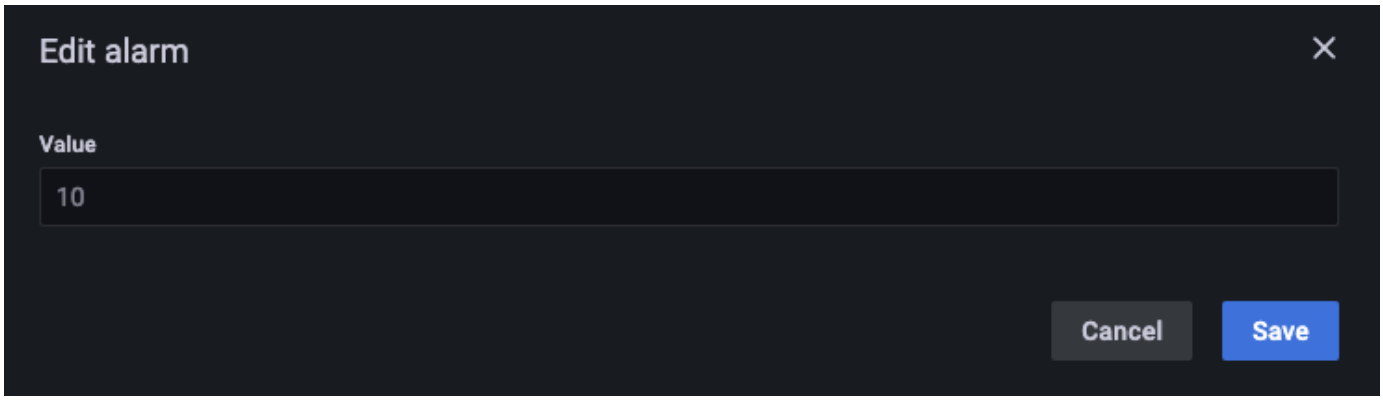
   > **Note**
   >
   > 공개 미리 보기의 경우 알림이 표시되지 않습니다. 경보 상태 및 임계값을 검토하여 속성이 올바르게 적용되었는지 확인해야 합니다.

7. 최신 값이 표시되도록 기본 쿼리 순서 오름차순을 사용해야 합니다.
8. 쿼리의 필터 섹션은 비워 둘 수 있습니다. 전체 구성은 아래 그림에 나와 있습니다.



9. 경보 편집 버튼을 사용하면 현재 경보 임계값을 변경하는 대화 상자를 불러올 수 있습니다.

10. 저장을 선택하여 새 임계값을 설정합니다.



> **Note**
>
> 이 패널은 현재가 포함된 실시간 시간 범위에서만 사용해야 합니다. 과거로 끝나고 시작되는 시간 범위와 함께 사용하면 경보 임계값을 항상 현재 임계값으로 편집하는 경우 예상치 못한 값이 표시될 수 있습니다.

AWS IoT TwinMaker Matterport 통합

Matterport는 실제 환경을 스캔하고 Matterport 디지털 트윈이라고도 하는 몰입형 3D 모델을 만들 수 있는 다양한 캡처 옵션을 제공합니다. 이러한 모델을 Matterport 스페이스라고 합니다. AWS IoT TwinMaker Matterport 통합을 지원하므로 Matterport 디지털 트윈을 AWS IoT TwinMaker 장면으로 가져올 수 있습니다. Matterport 디지털 트윈을와 페어링 AWS IoT TwinMaker하면 가상 환경에서 디지털 트윈 시스템을 시각화하고 모니터링할 수 있습니다.

Matterport 사용에 대한 자세한 내용은 Matterport의 문서AWS IoT TwinMaker 및 Matterport 페이지를 참조하십시오.

통합 주제

- 통합 개요
- Matterport 통합 사전 요구 사항
- Matterport 보안 인증을 생성하고 기록하십시오.
- 에 Matterport 자격 증명 저장 AWS Secrets Manager
- Matterport 공간을 AWS IoT TwinMaker 장면으로 가져오기
- AWS IoT TwinMaker Grafana 대시보드에서 Matterport 스페이스 사용
- AWS IoT TwinMaker 웹 애플리케이션에서 Matterport 스페이스 사용

통합 개요

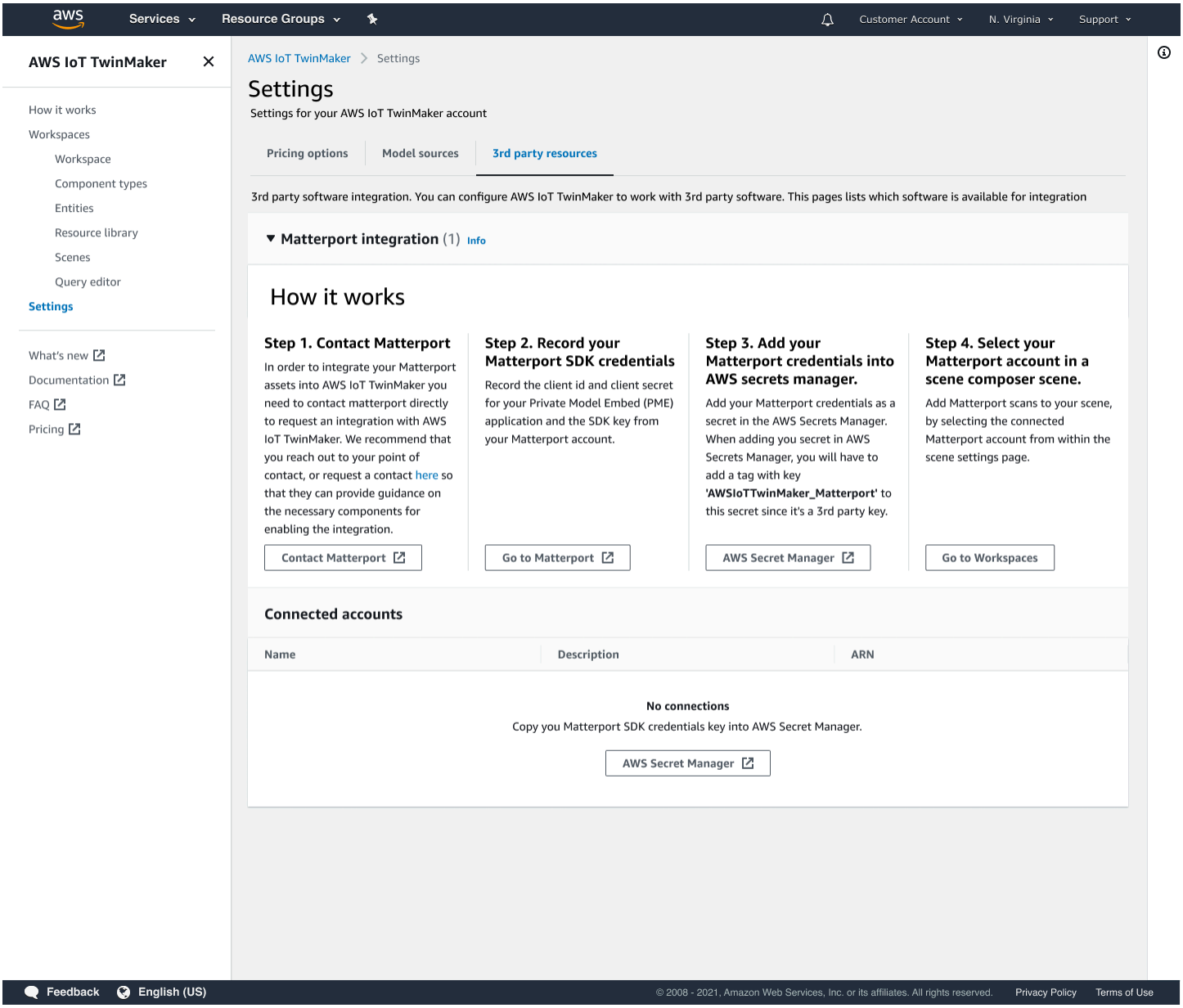
이 통합 기능을 사용하여 다음을 수행할 수 있습니다.

- AWS IoT TwinMaker 앱 키트에서 Matterport 태그와 공백을 사용합니다.
- AWS IoT TwinMaker Grafana 대시보드에서 가져온 Matterport 데이터를 봅니다. AWS IoT TwinMaker 및 Grafana 사용에 대한 자세한 내용은 Grafana 대시보드 통합 설명서를 참조하십시오.
- Matterport 공간을 AWS IoT TwinMaker 장면으로 가져옵니다.
- AWS IoT TwinMaker 장면의 데이터에 바인딩하려는 Matterport 태그를 선택하고 가져옵니다.
- AWS IoT TwinMaker 장면에서 Matterport 공간 및 태그 변경 사항을 자동으로 표시하고 동기화할 항목을 승인합니다.

통합 프로세스는 3개의 중요한 단계로 구성되어 있습니다.

1. Matterport 보안 인증을 생성하고 기록하십시오.
2. 에 Matterport 자격 증명 저장 AWS Secrets Manager
3. Matterport 공간을 AWS IoT TwinMaker 장면으로 가져오기

AWS IoT TwinMaker 콘솔에서 통합을 시작합니다. 콘솔의 설정 페이지에서 타사 리소스 아래에서 Matterport 통합을 열어 통합에 필요한 다양한 리소스 사이를 탐색할 수 있습니다.



Matterport 통합 사전 요구 사항

Matterport를와 통합하기 전에 다음 사전 조건을 충족하는지 AWS IoT TwinMaker 확인하세요.

- Enterprise-level Matterport 계정과 AWS IoT TwinMaker 통합에 필요한 Matterport 제품을 구매했습니다.
- AWS IoT TwinMaker 워크스페이스가 있습니다. 자세한 내용은 시작하기를 참조하세요 AWS IoT TwinMaker.
- AWS IoT TwinMaker 워크스페이스 역할을 업데이트했습니다. 작업 영역 역할을 만드는 방법에 대한 자세한 내용은 AWS IoT TwinMaker을(를) 위한 서비스 역할 만들기 및 관리를 참조하십시오.

다음을 작업 영역 역할에 추가합니다.

```
{
  "Effect": "Allow",
  "Action": "secretsmanager:GetSecretValue",
  "Resource": [
   "AWS Secrets Manager secret ARN"
  ]
}
```

- 통합을 활성화하는 데 필요한 라이선스를 구성하려면 Matterport에 문의해야 합니다. 또한 Matterport는 통합을 위한 Private Model Embed (PME)를 활성화합니다.

  이미 Matterport 계정 관리자가 있다면 직접 문의하십시오.

  Matterport 연락처가 없는 경우 다음 절차를 사용하여 Matterport에 연락하고 통합을 요청하십시오.

  1. Matterport 및 AWS IoT TwinMaker 페이지를 엽니다.
  2. 문의하기 버튼을 눌러 연락처양식을 엽니다.
  3. 양식에 필수 정보를 입력합니다.
  4. 준비가 되면 SAY HELLO를 선택하여 Matterport에 요청을 보내십시오.

  통합을 요청하면 통합 프로세스를 계속하는 데 필요한 Matterport SDK 및 Private Model Embed(PME) 보안 인증을 생성할 수 있습니다.

  > **Note**
  >
  > 이로 인해 신제품 또는 서비스 구매 수수료가 발생할 수 있습니다.

Matterport 보안 인증을 생성하고 기록하십시오.

Matterport를와 통합하려면 Matterport 자격 증명과 함께 AWS IoT TwinMaker를 제공해야 AWS Secrets Manager 합니다. 다음 절차를 사용하여 Matterport SDK 보안 인증을 생성하십시오.

1. Matterport 계정에 로그인하십시오.
2. 계정 설정 페이지로 이동합니다.
3. 설정 페이지에서 개발자 도구 옵션을 선택합니다.

4. 개발자 도구 페이지에서 SDK 키 관리 섹션으로 이동합니다.
5. SDK 키 관리 섹션에서 새 SDK 키를추가하는 옵션을 선택합니다.
6. Matterport SDK 키가 있으면 AWS IoT TwinMaker 및 Grafana 서버의 키에 도메인을 추가합니다. AWS IoT TwinMaker 앱 키트를 사용하는 경우 사용자 지정 도메인도 추가해야 합니다.
7. 다음으로 애플리케이션 통합 관리 섹션을 찾으면 PME 애플리케이션이 나열된 것을 확인할 수 있습니다. 다음의 정보를 기록합니다.

   - 클라이언트 ID
   - 클라이언트 비밀

   > **Note**
   >
   > 클라이언트 비밀은 한 번만 제공되므로 클라이언트 비밀을 기록해 두는 것이 좋습니다. Matterport 통합을 계속하려면 AWS Secrets Manager 콘솔에 클라이언트 비밀을 제시해야 합니다.

   이러한 보안 인증 정보는 필요한 구성 요소를 구매하고 Matterport에서 계정의 PME를 활성화하면 자동으로 생성됩니다. 이러한 보안 인증 정보가 나타나지 않으면 Matterport에 문의하십시오. 연락을 요청하려면 Matterport 및 AWS IoT TwinMaker연락처 양식을 참조하십시오.

Matterport SDK 보안 인증에 대한 자세한 내용은 Matterport의 공식 SDK 문서 SDK 문서 개요를 참조하십시오.

에 Matterport 자격 증명 저장 AWS Secrets Manager

다음 절차에 따라 Matterport 자격 증명을에 저장합니다 AWS Secrets Manager.

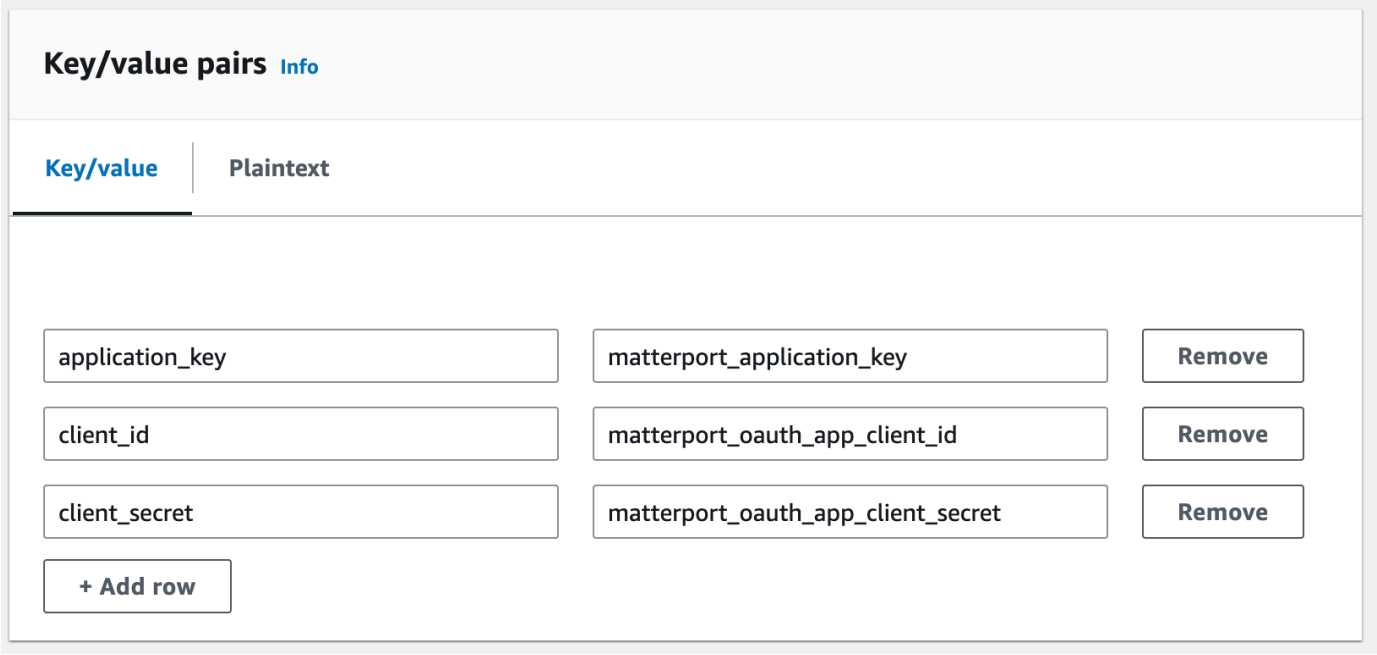
> **Note**
>
> Matterport 통합을 계속하려면 Matterport 보안 인증을 생성하고 기록하십시오. 주제의 프로시저에서 생성한 클라이언트 시크릿이 필요합니다.

1. AWS Secrets Manager 콘솔에 로그인합니다.

2. 시크릿 페이지로 이동한 다음 새 비밀 저장을 선택합니다.
3. 비밀 유형에서 다른 유형의 암호를 선택합니다.
4. 키/값 쌍 섹션에서 Matterport 보안 인증을 값으로 사용하여 다음 키-값 쌍을 추가하십시오.

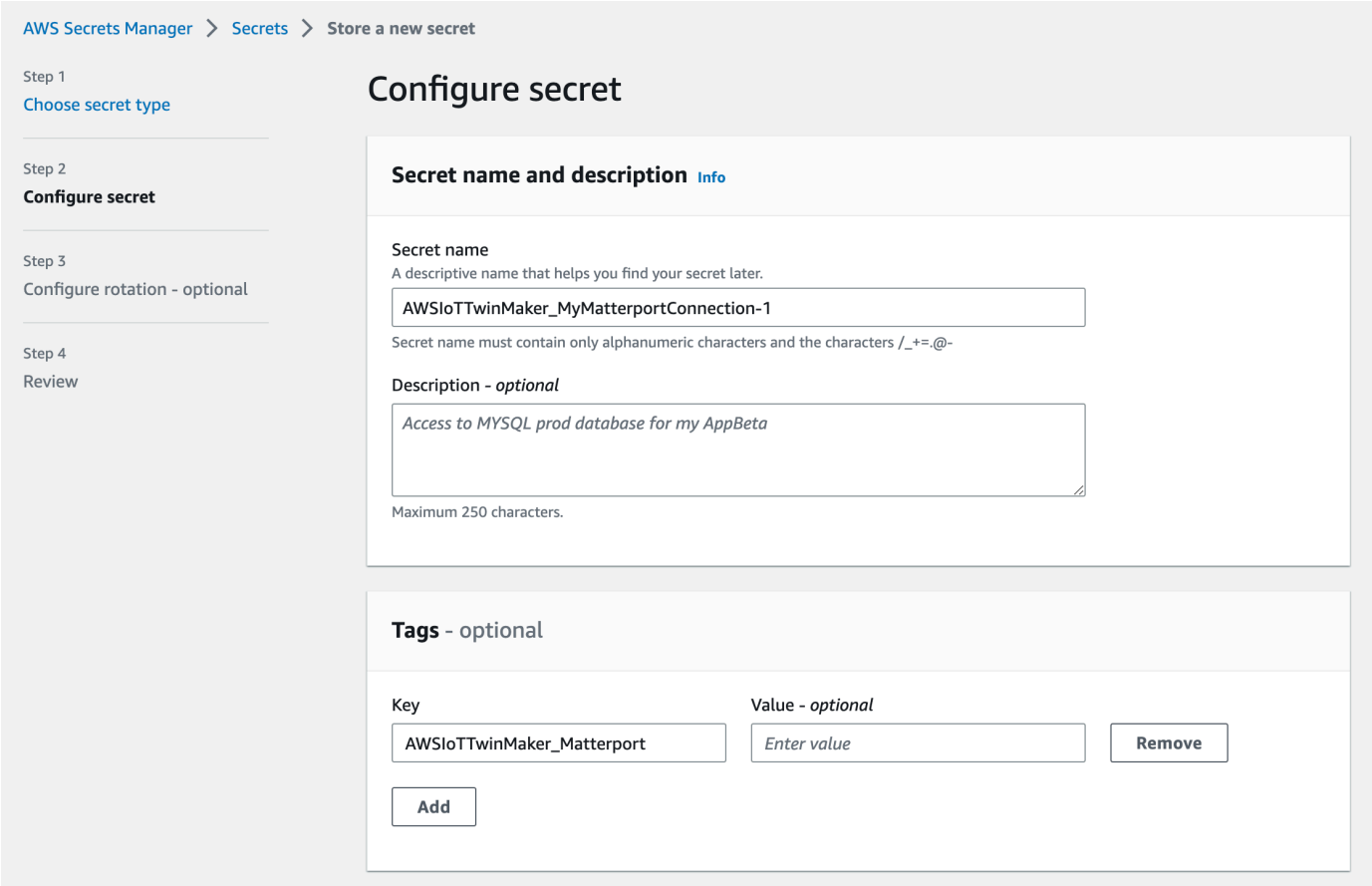
   - Key: `application_key` 및 Value:*<your Matterport credentials>*을(를) 사용하여 키-값 쌍을 생성합니다.
   - Key: `client_id` 및 Value:*<your Matterport credentials>*을(를) 사용하여 키-값 쌍을 생성합니다.
   - Key: `client_secret` 및 Value:*<your Matterport credentials>*을(를) 사용하여 키-값 쌍을 생성합니다.

   구성이 완료되면 다음 예와 비슷한 구성이 나타나는 것을 가질 수 있습니다.



5. 암호 키의 경우, 기본 암호 키`aws/secretsmanager`가 선택된 상태로 놔둘 수 있습니다.
6. 다음을 선택하여 비밀 구성 페이지로 이동합니다.
7. 비밀 이름 및 설명 필드를 입력합니다.
8. 태그 섹션에서 이 비밀에 태그를 추가하십시오.

   태그를 생성할 때 다음 스크린샷과 `AWSIoTTwinMaker_Matterport` 같이 키를 할당합니다.



> **Note**
>
> 태그를 추가해야 합니다. 태그가 선택 사항으로 나열되어 있더라도 타사 비밀을 AWS Secrets Manager로 추가할 때, 태그가 필요합니다.

값 필드는 선택 사항입니다. 일단 키를 제공한 후에는 추가를 선택하여 다음 단계로 넘어갈 수 있습니다.

9. 다음을 선택하여 로테이션 구성페이지로 이동합니다. 비밀 로테이션 설정은 선택 사항입니다. 비밀 추가를 완료하고 로테이션이 필요하지 않은 경우, 다음을 다시 선택합니다. 보안 암호 교체에 대한 자세한 내용은 AWS Secrets Manager 보안 암호 교체를 참조하세요.
10. 검토 페이지에서 비밀 구성을 확인하십시오. 비밀을 추가할 준비가 되면 스토어를 선택하십시오.

사용에 대한 자세한 내용은 다음 AWS Secrets Manager 설명서를 AWS Secrets Manager참조하세요.

- 를 사용하여 보안 암호 생성 및 관리 AWS Secrets Manager

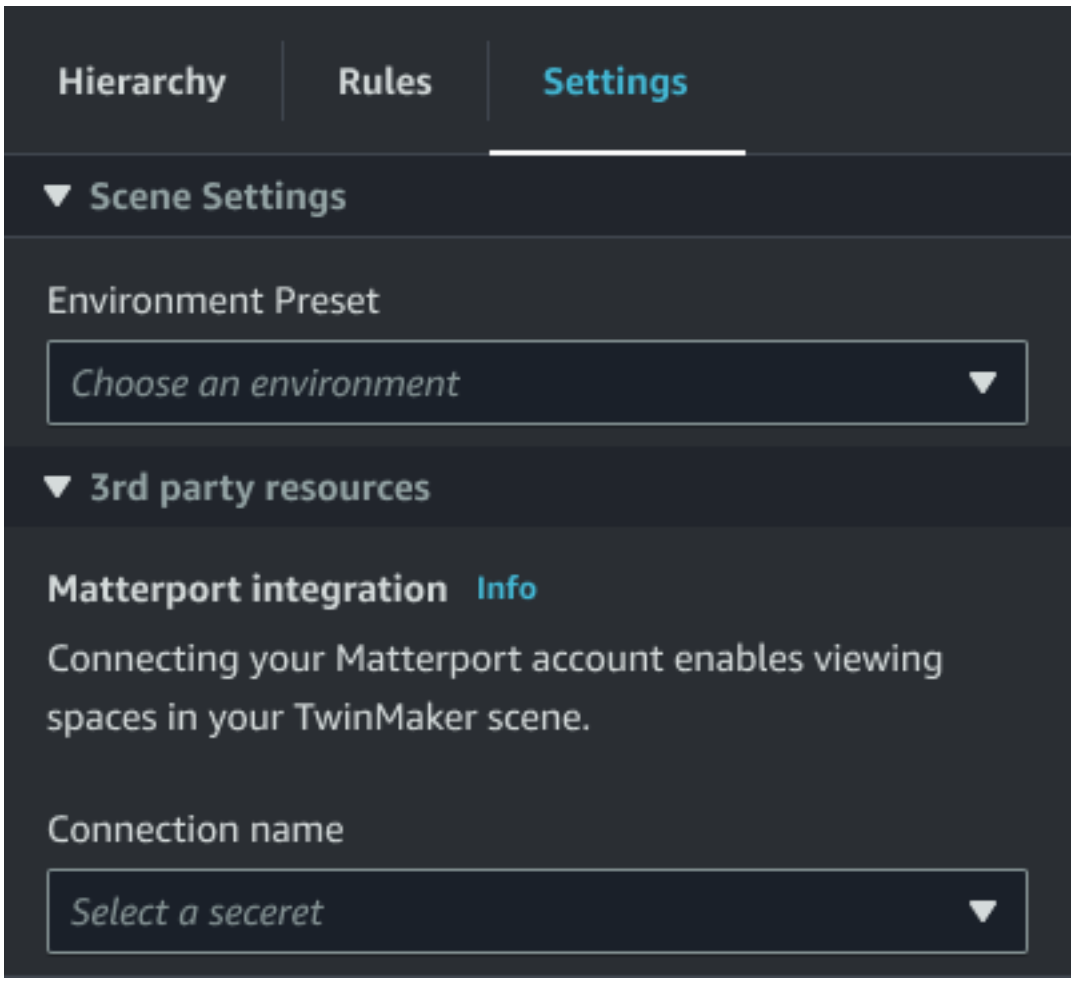
- 란 무엇입니까 AWS Secrets Manager?
- AWS Secrets Manager 보안 암호 교체

이제 Matterport 자산을 AWS IoT TwinMaker 장면으로 가져올 준비가 되었습니다. 다음 섹션 Matterport 공간을 AWS IoT TwinMaker 장면으로 가져오기의 절차를 참조하십시오.

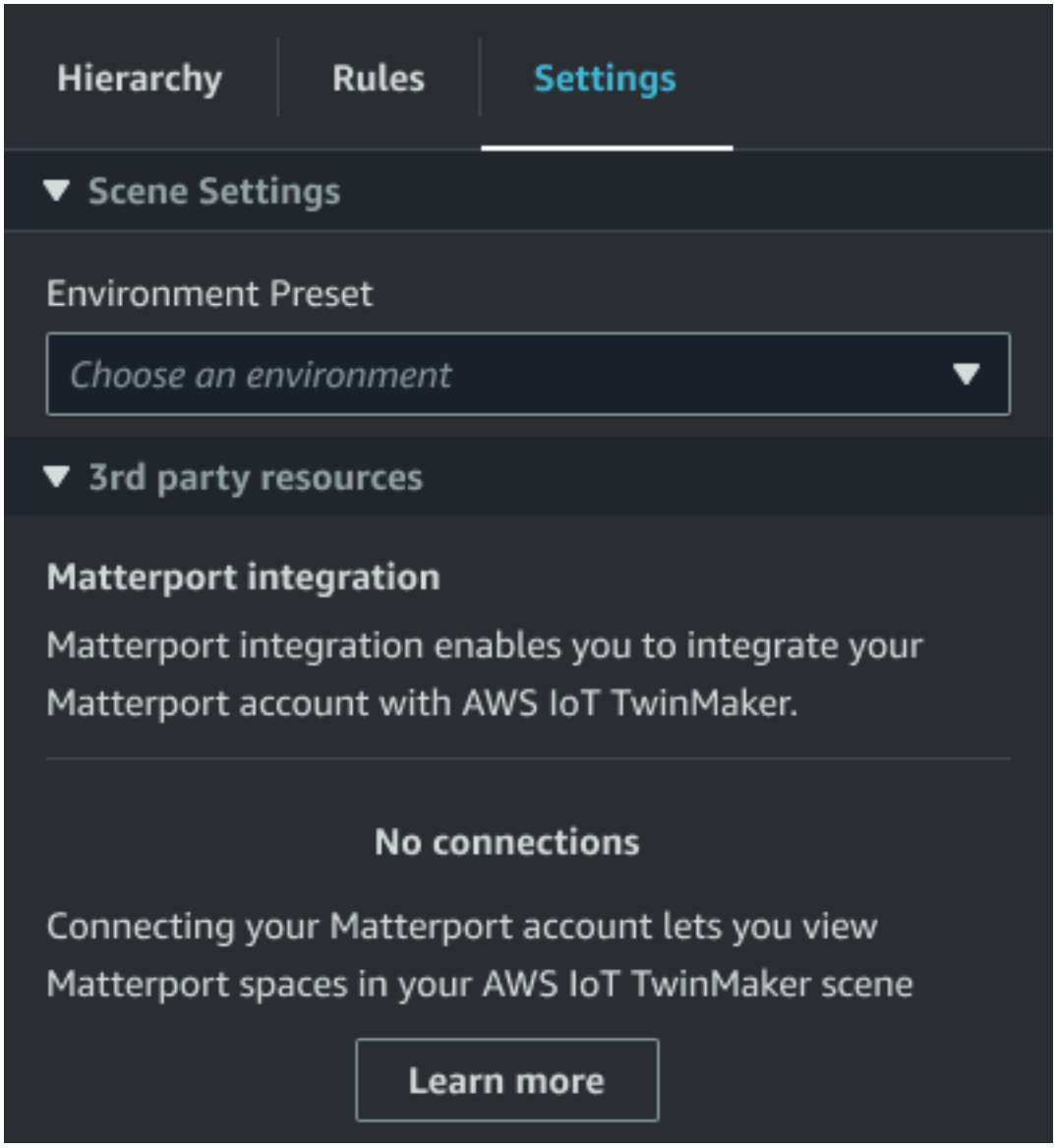
Matterport 공간을 AWS IoT TwinMaker 장면으로 가져오기

장면 설정 페이지 내에서 연결된 Matterport 계정을 선택하여 장면에 Matterport 스캔을 추가하십시오. Matterport 스캔 및 태그를 가져오려면 다음 절차를 따르십시오.

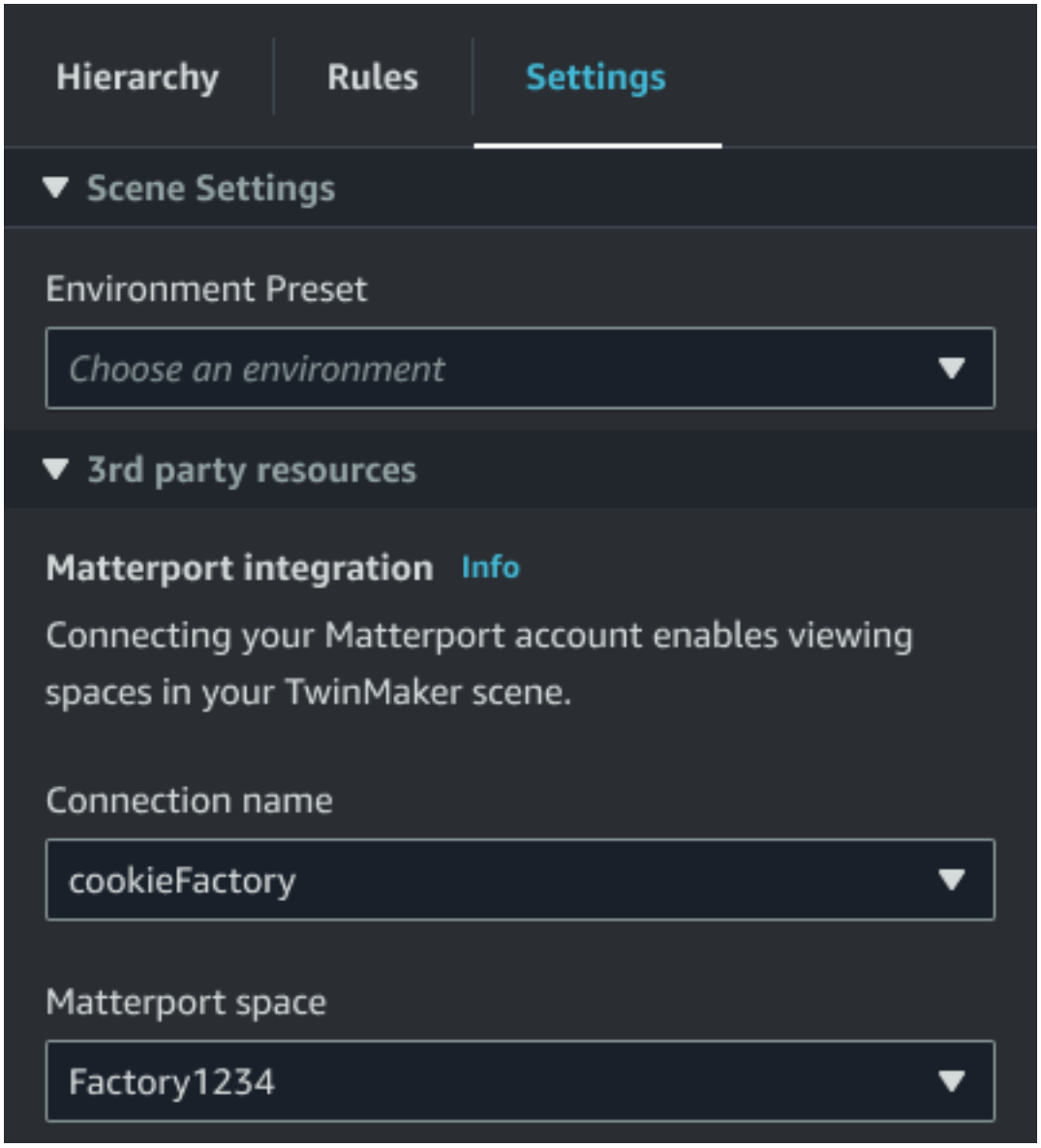
1. AWS IoT TwinMaker 콘솔에 로그인합니다.
2. Matterport 공간을 사용할 기존 AWS IoT TwinMaker 장면을 생성하거나 엽니다.
3. 장면이 열리면 설정 탭으로 이동합니다.
4. 설정에서 타사 리소스에서 연결 이름을 찾아서, 에 Matterport 자격 증명 저장 AWS Secrets Manager의 절차에서 생성한 비밀을 입력합니다.



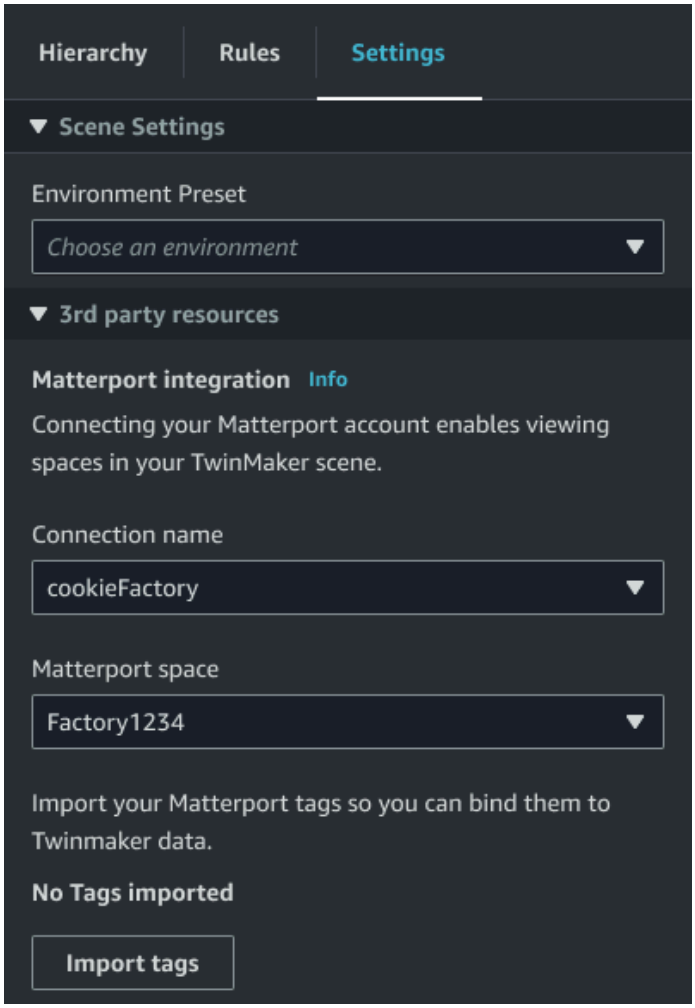
> **Note**
>
> 연결 없음이라는 메시지가 표시되면 AWS IoT TwinMaker 콘솔 설정 페이지로 이동하여 Matterport 통합 프로세스를 시작하십시오.



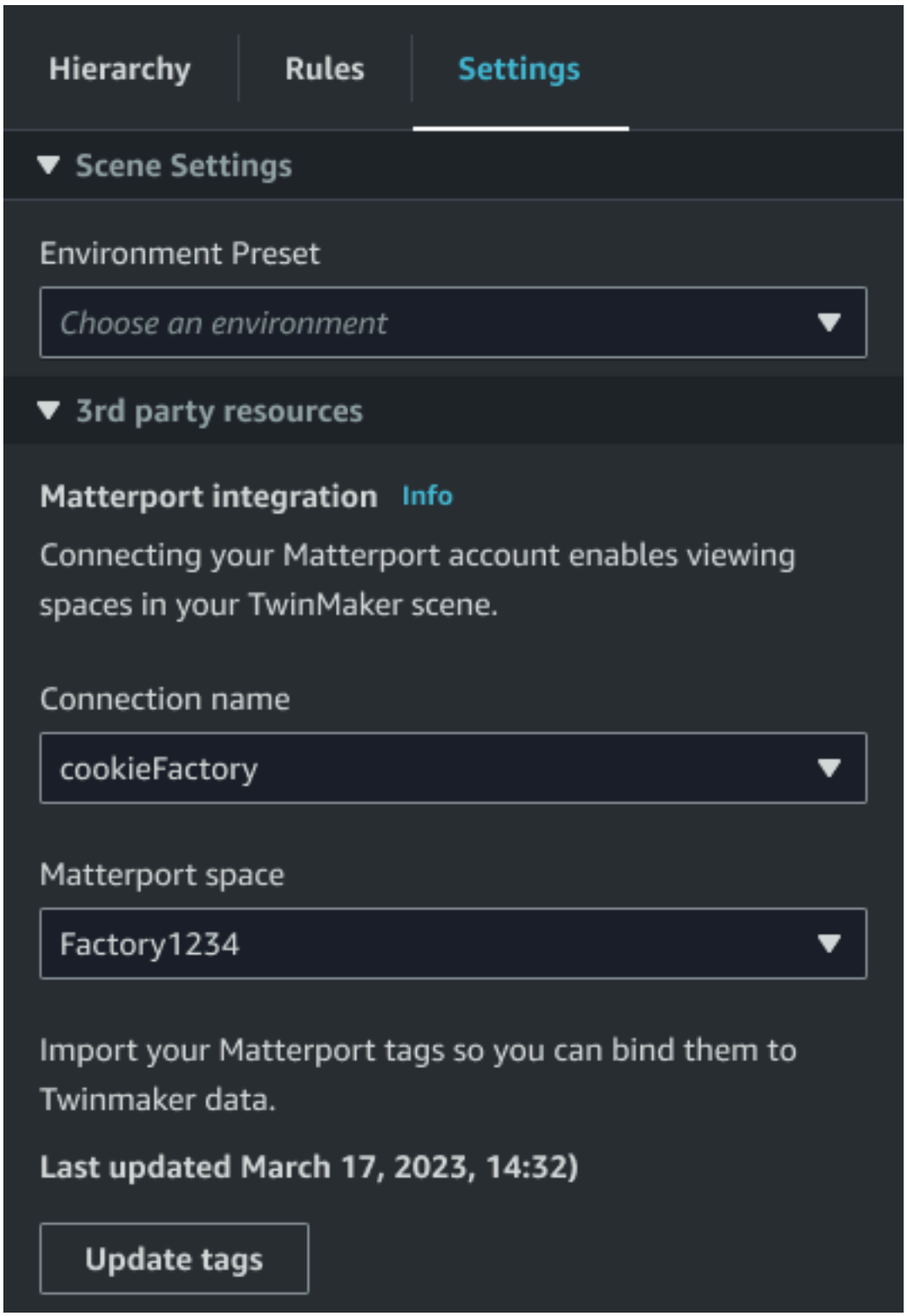
5. 다음으로, Matterport 스페이스 드롭다운에서 해당 공간을 선택하여 장면에서 사용하려는 Matterport 스페이스를 선택합니다.



6. 스페이스를 선택한 후 태그 가져오기 버튼을 눌러 Matterport 태그를 가져와 AWS IoT TwinMaker 장면 태그로 변환할 수 있습니다.



Matterport 태그를 가져오면 버튼이 태그 업데이트 버튼으로 대체됩니다. 항상 Matterport 계정의 최신 변경 사항을 반영 AWS IoT TwinMaker 하도록에서 Matterport 태그를 지속적으로 업데이트할 수 있습니다.



7. Matterport AWS IoT TwinMaker 와 성공적으로 통합되었으며 이제 AWS IoT TwinMaker 장면에 가져온 Matterport 공간과 태그가 모두 있습니다. 다른 장면과 마찬가지로이 AWS IoT TwinMaker 장면 내에서 작업할 수 있습니다.

AWS IoT TwinMaker 장면 작업에 대한 자세한 내용은 AWS IoT TwinMaker 장면 생성 및 편집을 참조하세요.

AWS IoT TwinMaker Grafana 대시보드에서 Matterport 스페이스 사용

Matterport 공간을 AWS IoT TwinMaker 장면으로 가져온 후에는 Grafana 대시보드에서 Matterport 공간을 사용하여 해당 장면을 볼 수 있습니다. Grafana를 로 이미 구성한 경우 Grafana 대시보드를 열어 가져온 Matterport 공간으로 장면을 볼 AWS IoT TwinMaker수 있습니다.

아직 Grafana AWS IoT TwinMaker 로를 구성하지 않은 경우 먼저 Grafana 통합 프로세스를 완료합니다. Grafana AWS IoT TwinMaker 와 통합할 때 두 가지 옵션을 선택할 수 있습니다. 자체 관리형 Grafana 인스턴스를 사용하거나 Amazon Managed Grafana를 사용할 수 있습니다.

Grafana 옵션 및 통합 프로세스에 대해 자세히 알아보려면 다음 설명서를 참조하십시오.

- AWS IoT TwinMaker Grafana 대시보드 통합.
- Amazon Managed Grafana
- 자체 관리형 Grafana.

AWS IoT TwinMaker 웹 애플리케이션에서 Matterport 스페이스 사용

Matterport 공간을 AWS IoT TwinMaker 장면으로 가져오면 AWS IoT 앱 키트 웹 애플리케이션에서 Matterport 공간을 사용하여 해당 장면을 볼 수 있습니다.

AWS IoT 애플리케이션 키트 사용에 대한 자세한 내용은 다음 설명서를 참조하세요.

- AWS IoT 앱 키트와 AWS IoT TwinMaker 함께를 사용하는 방법에 대한 자세한 내용은 섹션을 참조하세요AWS IoT TwinMaker UI 컴포넌트를 사용하여 사용자 지정 웹 애플리케이션 만들기.
- AWS IoT 애플리케이션 키트 사용에 대한 자세한 내용은 AWS IoT 애플리케이션 키트 Github 페이지를 참조하세요.
- AWS IoT 애플리케이션 키트를 사용하여 새 웹 애플리케이션을 시작하는 방법에 대한 지침은 공식 IoT 앱 키트 설명서 페이지를 참조하세요.

AWS IoT TwinMaker 비디오 통합

동영상 카메라는 디지털 트윈 시뮬레이션을 위한 좋은 기회를 제공합니다. AWS IoT TwinMaker 을(를) 사용하여 카메라의 위치와 물리적 조건을 시뮬레이션할 수 있습니다. 에서 현장 카메라 AWS IoT TwinMaker 에 대한 개체를 생성하고 비디오 구성 요소를 사용하여 사이트에서 AWS IoT TwinMaker 장면 또는 Grafana 대시보드로 라이브 비디오 및 메타데이터를 스트리밍합니다.

AWS IoT TwinMaker 는 두 가지 방법으로 엣지 디바이스에서 비디오를 캡처할 수 있습니다. Kinesis 동영상 스트림용 엣지 커넥터를 사용하여 엣지 디바이스에서 동영상을 스트리밍하거나, 엣지 디바이스에 동영상을 저장하고 MQTT 메시지로 동영상 업로드를 시작할 수 있습니다. 이 구성 요소를 사용하여 디바이스에서 비디오 데이터를 스트리밍하여 AWS IoT 서비스와 함께 사용할 수 있습니다. 필요한 리소스를 생성하고 Kinesis Video Streams용 엣지 커넥터를 배포하려면 GitHub의 Kinesis 동영상 스트림용 엣지 커넥터 시작하기를 참조하십시오. AWS IoT Greengrass 구성 요소에 대한 자세한 내용은 Kinesis Video Streams의 엣지 커넥터에 대한 AWS IoT Greengrass 설명서를 참조하세요. https://docs.aws.amazon.com/greengrass/v2/developerguide/kvs-edge-connector-component.html

필요한 AWS IoT SiteWise 모델을 생성하고 Kinesis Video Streams Greengrass 구성 요소를 구성한 후 AWS IoT TwinMaker 콘솔에서 엣지의 비디오를 디지털 트윈 애플리케이션으로 스트리밍하거나 녹화할 수 있습니다. Grafana 대시보드에서 디바이스의 라이브스트림 및 메타데이터를 볼 수도 있습니다. Grafana 및 통합에 대한 자세한 내용은 섹션을 AWS IoT TwinMaker참조하세요AWS IoT TwinMaker Grafana 대시보드 통합.

Kinesis 비디오 스트림용 엣지 커넥터를 사용하여에서 비디오 스트리밍 AWS IoT TwinMaker

Kinesis 비디오 스트림용 엣지 커넥터를 사용하면 AWS IoT TwinMaker 장면의 개체로 비디오 및 데이터를 스트리밍할 수 있습니다. 동영상 구성 요소를 사용하여 이 작업을 수행할 수 있습니다. 장면에 사용할 동영상 구성 요소를 생성하려면 다음 절차를 완료하십시오.

사전 조건

AWS IoT TwinMaker 장면에서 비디오 구성 요소를 생성하기 전에 다음 사전 조건을 완료했는지 확인합니다.

- Kinesis 비디오 스트림용 엣지 커넥터에 필요한 AWS IoT SiteWise 모델 및 자산을 생성했습니다. 커넥터용 AWS IoT SiteWise 자산 생성에 대한 자세한 내용은 Kinesis 동영상 스트림용 엣지 커넥터 시작하기를 참조하십시오.

- AWS IoT Greengrass 디바이스에 Kinesis 비디오 스트림 엣지 커넥터를 배포했습니다. Kinesis 동영상 스트림 엣지 커넥터 구성 요소 배포에 대한 자세한 내용은 배포 README를 참조하십시오.

AWS IoT TwinMaker 장면을 위한 비디오 구성 요소 생성

다음 단계를 완료하여 장면의 Kinesis 동영상 스트림 구성 요소용 엣지 커넥터를 생성하십시오.

1. AWS IoT TwinMaker 콘솔에서 비디오 구성 요소를 추가할 장면을 엽니다.
2. 장면이 열리면 기존 개체를 선택하거나 구성 요소를 추가할 개체를 만든 다음 구성 요소 추가를 선택합니다.
3. 구성 요소 추가 창에서 구성 요소 이름을 입력하고 유형으로는 com.amazon.iotsitewise.connector.edgevideo를 선택합니다.
4. 생성한 AWS IoT SiteWise 카메라 모델의 이름을 선택하여 자산 모델을 선택합니다. 이름은 `EdgeConnectorForKVSCameraModel-0abc` 형식을 취해야 합니다. 즉 끝에 있는 문자와 숫자로 구성된 문자열이 본인의 자산 이름과 일치해야 합니다.
5. 자산에서 비디오를 스트리밍할 AWS IoT SiteWise 카메라 자산을 선택합니다. 현재 동영상 스트림의 미리보기를 보여주는 작은 창이 나타납니다.

   > **Note**
   >
   > 동영상 스트리밍을 테스트하려면 테스트를 선택합니다. 이 테스트에서는 MQTT 이벤트를 전송하여 동영상 라이브 스트리밍을 시작합니다. 플레이어에 동영상이 표시될 때까지 잠시 기다려 주십시오.

6. 개체에 동영상 구성 요소를 추가하려면 구성 요소 추가를 선택합니다.

Kinesis 동영상 스트림의 동영상 및 메타데이터를 Grafana 대시보드에 추가

AWS IoT TwinMaker 장면에서 개체에 대한 비디오 구성 요소를 생성한 후 Grafana에서 비디오 패널을 구성하여 라이브 스트림을 볼 수 있습니다. Grafana AWS IoT TwinMaker 와 올바르게 통합되었는지 확인합니다. 자세한 내용은 AWS IoT TwinMaker Grafana 대시보드 통합 단원을 참조하십시오.

> **Important**
>
> Grafana 대시보드에서 비디오를 보려면 Grafana 데이터 소스에 적절한 IAM 권한이 있는지 확인해야 합니다. 필요한 역할 및 정책을 생성하려면 대시보드 IAM 역할 생성을(를) 참조하십시오.

Grafana 대시보드에서 Kinesis Video Streams와 메타데이터를 보려면 다음 단계를 완료하십시오.

1. AWS IoT TwinMaker 대시보드를 엽니다.
2. 패널 추가를 선택한 다음 빈 패널 추가를 선택합니다.

> **Note**
>
> Grafana v10.4의 경우 AWS IoT TwinMaker 비디오 플레이어는 위젯에서 찾을 수 있습니다. 추가 >> 위젯을 선택합니다.

3. 패널 목록에서 AWS IoT TwinMaker 동영상 플레이어 패널을 선택합니다.
4. AWS IoT TwinMaker 동영상 플레이어 패널에서 KinesisVideoStreamName의 스트림 이름과 동영상을 스트리밍하려는 Kinesis 동영상 스트림의 이름을 입력합니다.

> **Note**
>
> 메타데이터를 Grafana 동영상 패널로 스트리밍하려면 먼저 동영상 스트리밍 구성 요소가 있는 개체를 만들어야 합니다.

5. 선택 사항: AWS IoT SiteWise 자산에서 비디오 플레이어로 메타데이터를 스트리밍하려면 개체에서 AWS IoT TwinMaker 장면에서 생성한 AWS IoT TwinMaker 개체를 선택합니다. 구성 요소 이름으로는 AWS IoT TwinMaker 장면의 개체에 대해 생성한 동영상 구성 요소를 선택합니다

AWS IoT TwinMaker 플링크 라이브러리 사용하기

AWS IoT TwinMaker은(는) 디지털 트윈에 사용되는 외부 데이터 저장소에 데이터를 읽고 쓸 때 사용할 수 있는 Flink 라이브러리를 제공합니다

AWS IoT TwinMaker Flink 라이브러리를 Amazon Managed Service for Apache Flink에 사용자 지정 커넥터로 설치하고 Amazon Managed Service for Apache Flink의 Zeppelin 노트북에서 Flink SQL 쿼리를 수행하여 Flink 라이브러리를 사용할 수 있습니다. 노트북은 지속적으로 실행되는 스트림 처리 애플리케이션으로 승격될 수 있습니다. 라이브러리는 AWS IoT TwinMaker 구성 요소를 활용하여 작업 영역에서 데이터를 검색합니다.

AWS IoT TwinMaker Flink 라이브러리에는 다음이 필요합니다.

사전 조건

1. 장면과 구성 요소로 완전히 채워진 작업 영역. AWS 서비스 (AWS IoT SiteWise 및 Kinesis 동영상 스트림)의 데이터에 내장된 구성 요소 유형을 사용하세요. 타사 소스의 데이터에 대한 사용자 지정 구성 요소 유형을 생성하세요. 자세한 내용은 ???을(를) 참조하세요.
2. Managed Service for Apache Flink for Apache Flink가 포함된 Studio 노트북에 대한 이해. 이 노트북은 Apache Zeppelin으로 구동되며 Apache Flink 프레임워크를 사용합니다. 자세한 내용은 Managed Service for Apache Flink for Apache Flink가 포함된 Studio 노트북 사용하기를 참조하세요.

라이브러리 사용에 대한 지침은 AWS IoT TwinMakerFlink 라이브러리 사용 설명서를 참조하세요.

AWS IoT TwinMaker샘플에서 빠른 시작으로 AWS IoT TwinMaker을(를) 설정하는 방법에 대한 지침은 샘플 통찰력 애플리케이션의 README 파일을 참조하세요.

에서 로깅 및 모니터링 AWS IoT TwinMaker

모니터링은 AWS IoT TwinMaker 및 다른 AWS 솔루션의 안정성, 가용성 및 성능을 유지하는 데 중요한 부분입니다.는 서비스를 모니터링하고, 이상이 있을 때 보고하고, 필요한 경우 자동 조치를 취할 수 있도록 다음 모니터링 도구를 AWS IoT TwinMaker 지원합니다.

- Amazon CloudWatch는 AWS 리소스와 실행 중인 애플리케이션을 실시간으로 모니터링합니다 AWS. 지표를 수집 및 추적하고, 사용자 지정 대시보드를 생성할 수 있으며, 지정된 지표가 지정한 임계값에 도달하면 사용자에게 알리거나 조치를 취하도록 경보를 설정할 수 있습니다. 예를 들어 CloudWatch에서 Amazon EC2 인스턴스의 CPU 사용량 또는 기타 지표를 추적하고 필요할 때 자동으로 새 인스턴스를 시작할 수 있습니다. 자세한 정보는 Amazon CloudWatch 사용 설명서를 참조하십시오.
- Amazon CloudWatch Logs는 AWS IoT TwinMaker 게이트웨이, CloudTrail 및 기타 소스의 로그 파일을 모니터링, 저장 및 액세스합니다. CloudWatch Logs는 로그 파일의 정보를 모니터링하고 특정 임계값에 도달하면 사용자에게 알릴 수 있습니다. 또한 매우 내구성이 뛰어난 스토리지에 로그 데이터를 저장할 수 있습니다. 자세한 내용은 Amazon CloudWatch Logs 사용자 안내서를 참조하세요.
- AWS CloudTrail는 AWS 계정에 의해 또는 계정을 대신하여 수행된 API 호출 및 관련 이벤트를 캡처하고 사용자가 지정한 Amazon S3 버킷에 로그 파일을 전송합니다. 호출한 사용자 및 계정 AWS, 호출이 수행된 소스 IP 주소, 호출이 발생한 시기를 식별할 수 있습니다. 자세한 내용은 AWS CloudTrail 사용 설명서를 참조하십시오.

주제

Amazon CloudWatch 지표 AWS IoT TwinMaker 를 사용한 모니터링

원시 데이터를 수집 AWS IoT TwinMaker 하여 읽기 가능하며 실시간에 가까운 지표로 처리하는 CloudWatch를 사용하여 모니터링할 수 있습니다. 이러한 통계는 15개월간 보관되므로 기록 정보에 액세스하고 웹 애플리케이션 또는 서비스가 어떻게 실행되고 있는지 전체적으로 더 잘 파악할 수 있습니다. 특정 임곗값을 주시하다가 해당 임곗값이 충족될 때 알림을 전송하거나 조치를 취하도록 경보를 설정할 수도 있습니다. 자세한 내용은 Amazon CloudWatch 사용 설명서를 참조하세요.

AWS IoT TwinMaker 는 다음 섹션에 나열된 지표와 차원을 `AWS/IoTTwinMaker` 네임스페이스에 게시합니다.

> **Tip**
>
> AWS IoT TwinMaker 는 1분 간격으로 지표를 게시합니다. CloudWatch 콘솔에서 이러한 지표를 그래프로 볼 때는 지표 데이터의 사용 가능한 최고 해상도를 보려면 1분의 기간을 선택하는 것이 좋습니다.

목차

Metrics

AWS IoT TwinMaker 는 다음 지표를 게시합니다.

Metrics

지표	설명
`ComponentTypeCreationFailure`	이 지표는 구성 요소 유형 생성의 성공 여부를 보고합니다. 지표는 구성 요소 유형이 `CREATING` 상태일 때 게시됩니다. 이는 스키마 이니셜라이저에서 필수 속성을 사용하여 구성 요소 유형을 생성하고 이러한 속성이 기본값으로 인스턴스화될 때 발생합니다. 지표 값은 성공 `0` 또는 실패 `1` 값일 수 있습니다. 크기: ComponentTypeId, WorkspaceId. 단위: 개
`ComponentTypeUpdateFailure`	이 지표는 구성 요소 유형 업데이트의 성공 여부를 보고합니다. 지표는 구성 요소 유형이 `UPDATING` 상태일 때 게시됩니다. 이는 구성 요소 유형이 스키마 이니

지표	설명
	셜라이저의 필수 속성으로 업데이트되고 이 속성이 기본값으로 인스턴스화될 때 발생합니다. 지표 값은 성공 0 또는 실패 1 값일 수 있습니다. 크기: ComponentTypeId, WorkspaceId. 단위: 개
EntityCreationFailure	이 지표는 개체 생성 성공 여부를 보고합니다. 지표는 개체가 CREATING 상태에 있을 때 게시됩니다. 이는 구성 요소를 사용하여 개체를 만들 때 발생합니다. 지표 값은 성공 0 또는 실패 1 값일 수 있습니다. 크기: EntityName, EntityId, WorkspaceId 단위: 개
EntityUpdateFailure	이 지표는 개체 업데이트의 성공 여부를 보고합니다. 지표는 개체가 UPDATING 상태에 있을 때 게시됩니다. 이는 개체가 업데이트될 때 발생합니다. 지표 값은 성공 0 또는 실패 1 값일 수 있습니다. 크기: EntityName, EntityId, WorkspaceId 단위: 개

지표	설명
`EntityDeletionFailure`	이 지표는 개체 삭제 성공 여부를 보고합니다. 지표는 개체가 `DELETING` 상태에 있을 때 게시됩니다. 이는 개체가 삭제될 때 발생합니다. 지표 값은 성공 `0` 또는 실패 `1` 값일 수 있습니다. 크기: EntityName, EntityId, WorkspaceId 단위: 개

> **Tip**
>
> 모든 지표는 `AWS/IoTTwinMaker` 네임스페이스에 게시됩니다.

를 사용하여 AWS IoT TwinMaker API 호출 로깅 AWS CloudTrail

AWS IoT TwinMaker 는 사용자 AWS CloudTrail, 역할 또는 서비스가 수행한 작업에 대한 레코드를 제공하는 AWS 서비스와 통합됩니다 AWS IoT TwinMaker. CloudTrail은 AWS IoT TwinMaker 에 대한 API 직접 호출을 이벤트로 캡처합니다. 캡처되는 호출에는 AWS IoT TwinMaker 콘솔의 호출과 AWS IoT TwinMaker API 작업에 대한 코드 호출이 포함됩니다. 추적을 생성하면 이벤트를 포함하여 CloudTrail 이벤트를 지속적으로 Amazon S3 버킷에 배포할 수 있습니다 AWS IoT TwinMaker. 추적을 구성하지 않은 경우에도 이벤트 기록에서 CloudTrail 콘솔의 최신 이벤트를 볼 수 있습니다. CloudTrail에서 수집한 정보를 사용하여 수행된 요청, 요청이 수행된 AWS IoT TwinMaker IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다.

CloudTrail에 대한 자세한 내용은 AWS CloudTrail 사용자 안내서를 참조하세요.

AWS IoT TwinMaker CloudTrail의 정보

AWS 계정을 생성하면 CloudTrail이 자동으로 활성화됩니다. CloudTrail 레코드는 이벤트 기록의 다른 AWS 서비스 이벤트와 AWS IoT TwinMaker함께에서 발생하는 이벤트 활동을 지원합니다. AWS 계정에서 최근 이벤트를 보고 검색하고 다운로드할 수 있습니다. 자세한 내용은 CloudTrail 이벤트 기록을 사용하여 이벤트 보기를 참조하세요.

에 대한 이벤트를 포함하여 AWS 계정에 이벤트를 지속적으로 기록하려면 추적을 AWS IoT TwinMaker생성합니다. CloudTrail은 추적을 사용하여 Amazon S3 버킷으로 로그 파일을 전송할 수 있습니다. 기본적으로 콘솔에서 추적을 생성하면 추적이 모든 AWS 리전에 적용됩니다. CloudTrail은 AWS 파티션의 모든 리전에서 이벤트를 로깅하고 지정한 Amazon S3 버킷으로 로그 파일을 전송합니다. 또한 CloudTrail 로그에서 수집된 이벤트 데이터를 추가로 분석하고 조치를 취하도록 다른 AWS 서비스를 구성할 수 있습니다. 자세한 내용은 다음 자료를 참조하세요.

- 추적 생성 개요
- CloudTrail 지원 서비스 및 통합
- CloudTrail에 대한 Amazon SNS 알림 구성
- 여러 리전에서 CloudTrail 로그 파일 수신 및 여러 계정에서 CloudTrail 로그 파일 수신

대부분의 AWS IoT TwinMaker 작업은 CloudTrail에서 로깅되며 AWS IoT TwinMaker API 참조에 문서화됩니다.

다음 데이터 영역 작업은 CloudTrail에 의해 로깅되지 않습니다.

- GetPropertyValue
- GetPropertyValueHistory
- BatchPutPropertyValues

모든 이벤트 또는 로그 항목에는 요청을 생성했던 사용자에 관한 정보가 포함됩니다. 보안 인증 정보를 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청을 루트로 했는지 아니면 사용자 자격 증명으로 했는지 여부
- 역할 또는 페더레이션 사용자에 대한 임시 보안 인증을 사용하여 요청이 생성되었는지 여부.
- 요청이 다른 AWS 서비스에서 이루어졌는지 여부입니다.

자세한 내용은 CloudTrail userIdentity 요소를 참조하십시오.

의 보안 AWS IoT TwinMaker

의 클라우드 보안 AWS 이 최우선 순위입니다. AWS 고객은 보안에 가장 민감한 조직의 요구 사항을 충족하도록 구축된 데이터 센터 및 네트워크 아키텍처의 이점을 누릴 수 있습니다.

보안은 AWS 와 사용자 간의 공동 책임입니다. 공동 책임 모델은 이 사항을 클라우드의 보안 및 클라우드 내 보안으로 설명합니다.

- 클라우드 보안 - AWS 는에서 AWS 서비스를 실행하는 인프라를 보호할 책임이 있습니다 AWS 클라우드. AWS 또한는 안전하게 사용할 수 있는 서비스를 제공합니다. 타사 감사자는 AWS 규정 준수 프로그램 일환으로 보안의 효과를 정기적으로 테스트하고 확인합니다. 에 적용되는 규정 준수 프로그램에 대한 자세한 내용은 규정 준수 프로그램 AWS 제공 범위 내 서비스규정 준수 프로그램 제공 범위 내 서비스를 AWS IoT TwinMaker참조하세요.
- 클라우드의 보안 - 사용자의 책임은 사용하는 AWS 서비스에 따라 결정됩니다. 또한 귀하는 귀사의 데이터 민감도, 귀사의 요구 사항, 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다.

이 설명서는를 사용할 때 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다 AWS IoT TwinMaker. 다음 주제에서는 보안 및 규정 준수 목표를 충족하도록 AWS IoT TwinMaker 를 구성하는 방법을 보여줍니다. 또한 AWS IoT TwinMaker 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스를 사용하는 방법을 알아봅니다.

주제

- 의 데이터 보호 AWS IoT TwinMaker
- 에 대한 자격 증명 및 액세스 관리 AWS IoT TwinMaker
- AWS IoT TwinMaker 및 인터페이스 VPC 엔드포인트(AWS PrivateLink)
- 에 대한 규정 준수 검증 AWS IoT TwinMaker
- 의 복원력 AWS IoT TwinMaker
- 의 인프라 보안 AWS IoT TwinMaker

의 데이터 보호 AWS IoT TwinMaker

AWS 공동 책임 모델 은 AWS IoT TwinMaker의 데이터 보호에 적용됩니다. 이 모델에 설명된 대로 AWS 는 모든를 실행하는 글로벌 인프라를 보호할 책임이 있습니다 AWS 클라우드. 사용자는 이 인프

라에 호스팅되는 콘텐츠에 대한 통제 권한을 유지할 책임이 있습니다. 사용하는 AWS 서비스 의 보안 구성과 관리 태스크에 대한 책임도 사용자에게 있습니다. 데이터 프라이버시에 대한 자세한 내용은 데이터 프라이버시 FAQ 참조하세요. 유럽의 데이터 보호에 대한 자세한 내용은 일반 데이터 보호 규정 (GDPR) 센터를 참조하세요.

데이터 보호를 위해 자격 증명을 보호하고 AWS 계정 AWS IAM Identity Center 또는 AWS Identity and Access Management (IAM)를 사용하여 개별 사용자를 설정하는 것이 좋습니다. 이렇게 하면 개별 사용자에게 자신의 직무를 충실히 이행하는 데 필요한 권한만 부여됩니다. 또한 다음과 같은 방법으로 데이터를 보호하는 것이 좋습니다.

- 각 계정에 다중 인증(MFA)을 사용합니다.
- SSL/TLS를 사용하여 AWS 리소스와 통신합니다. TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- 를 사용하여 API 및 사용자 활동 로깅을 설정합니다 AWS CloudTrail. CloudTrail 추적을 사용하여 AWS 활동을 캡처하는 방법에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 CloudTrail 추적 작업을 참조하세요.
- 내부의 모든 기본 보안 제어와 함께 AWS 암호화 솔루션을 사용합니다 AWS 서비스.
- Amazon S3에 저장된 민감한 데이터를 검색하고 보호하는 데 도움이 되는 Amazon Macie와 같은 고급 관리형 보안 서비스를 사용합니다.
- 명령줄 인터페이스 또는 API를 AWS 통해에 액세스할 때 FIPS 140-3 검증 암호화 모듈이 필요한 경우 FIPS 엔드포인트를 사용합니다. 사용 가능한 FIPS 엔드포인트에 대한 자세한 내용은 연방 정보 처리 표준(FIPS) 140-3을 참조하세요.

고객의 이메일 주소와 같은 기밀 정보나 중요한 정보는 태그나 이름 필드와 같은 자유 형식 텍스트 필드에 입력하지 않는 것이 좋습니다. 여기에는 AWS IoT TwinMaker 또는 기타 AWS 서비스 에서 콘솔, API AWS CLI또는 AWS SDKs를 사용하여 작업하는 경우가 포함됩니다. 이름에 사용되는 태그 또는 자유 형식 텍스트 필드에 입력하는 모든 데이터는 청구 또는 진단 로그에 사용될 수 있습니다. 외부 서버에 URL을 제공할 때 해당 서버에 대한 요청을 검증하기 위해 보안 인증 정보를 URL에 포함시켜서는 안 됩니다.

저장된 데이터 암호화

AWS IoT TwinMaker 는 선택한 경우 서비스가 생성하는 Amazon S3 버킷에 워크스페이스 정보를 저장합니다. 서비스가 사용자를 위해 생성하는 버킷에는 기본 서버 측 암호화가 활성화되어 있습니다. 새 작업 영역을 생성할 때 자체 Amazon S3 버킷을 사용하기로 선택한 경우, 기본 서버 측 암호화를 활성화하는 것이 좋습니다. Amazon S3 기본 암호화에 대한 자세한 내용은 Amazon S3 버킷에 대한 기본 서버 측 암호화 동작 설정을 참조하십시오.

전송 중 암호화

로 전송되는 모든 데이터는 HTTPS 프로토콜을 사용하여 TLS 연결을 통해 AWS IoT TwinMaker 전송되므로 전송 중에 기본적으로 안전합니다.

> **Note**
>
> 가 Amazon S3 버킷과 AWS IoT TwinMaker 상호 작용할 때 전송 중 암호화를 적용하려면 Amazon S3 버킷 주소에서 HTTPS를 제어로 사용하는 것이 좋습니다. Amazon S3 버킷에 대한 자세한 내용은 Amazon S3 버킷 을 가지고 하는 생성, 구성 및 작업을 참조하십시오.

에 대한 자격 증명 및 액세스 관리 AWS IoT TwinMaker

AWS Identity and Access Management (IAM)는 관리자가 AWS 리소스에 대한 액세스를 안전하게 제어하는 데 도움이 AWS 서비스 되는 입니다. IAM 관리자는 누가 AWS IoT TwinMaker 리소스를 사용할 수 있는 인증(로그인) 및 권한(권한 있음)을 받을 수 있는지 제어합니다. IAM은 추가 비용 없이 사용할 수 AWS 서비스 있는 입니다.

주제

- 대상
- ID를 통한 인증
- 정책을 사용하여 액세스 관리
- AWS IoT TwinMaker 에서 IAM을 사용하는 방법
- 에 대한 자격 증명 기반 정책 예제 AWS IoT TwinMaker
- AWS IoT TwinMaker 자격 증명 및 액세스 문제 해결
- 에 서비스 연결 역할 사용 AWS IoT TwinMaker
- AWS 에 대한 관리형 정책 AWS IoT TwinMaker

대상

AWS Identity and Access Management (IAM)를 사용하는 방법은 역할에 따라 다릅니다.

- 서비스 사용자 - 기능에 액세스할 수 없는 경우 관리자에게 권한 요청(참조AWS IoT TwinMaker 자격 증명 및 액세스 문제 해결)

- 서비스 관리자 - 사용자 액세스 결정 및 권한 요청 제출(AWS IoT TwinMaker 에서 IAM을 사용하는 방법 참조)
- IAM 관리자 - 액세스를 관리하기 위한 정책 작성(에 대한 자격 증명 기반 정책 예제 AWS IoT TwinMaker 참조)

ID를 통한 인증

인증은 자격 증명 자격 증명을 AWS 사용하여에 로그인하는 방법입니다. AWS 계정 루트 사용자, IAM 사용자 또는 IAM 역할을 수임하여 인증되어야 합니다.

AWS IAM Identity Center (IAM Identity Center), Single Sign-On 인증 또는 Google/Facebook 자격 증명과 같은 자격 증명 소스의 자격 증명을 사용하여 페더레이션 자격 증명으로 로그인할 수 있습니다. 로그인하는 방법에 대한 자세한 내용은 AWS 로그인 사용 설명서의 AWS 계정에 로그인하는 방법 섹션을 참조하세요.

프로그래밍 방식 액세스를 위해는 요청에 암호화 방식으로 서명할 수 있는 SDK 및 CLI를 AWS 제공합니다. 자세한 내용은 IAM 사용 설명서의 API 요청용AWS Signature Version 4 섹션을 참조하세요.

AWS 계정 루트 사용자

를 생성할 때 모든 AWS 서비스 및 리소스에 대한 완전한 액세스 권한이 있는 AWS 계정 theroot 사용자라는 하나의 로그인 자격 증명으로 AWS 계정시작합니다. 일상적인 태스크에 루트 사용자를 사용하지 않을 것을 강력히 권장합니다. 루트 사용자 자격 증명이 필요한 작업은 IAM 사용 설명서의 루트 사용자 자격 증명이 필요한 작업 섹션을 참조하세요.

페더레이션 ID

가장 좋은 방법은 인간 사용자에게 자격 증명 공급자와의 페더레이션을 사용하여 임시 자격 증명을 AWS 서비스 사용하여에 액세스하도록 요구하는 것입니다.

페더레이션 자격 증명은 엔터프라이즈 디렉터리, 웹 자격 증명 공급자 또는 자격 증명 소스의 자격 증명을 AWS 서비스 사용하여 Directory Service 에 액세스하는 사용자입니다. 페더레이션 ID는 임시 자격 증명을 제공하는 역할을 수임합니다.

중앙 집중식 액세스 관리를 위해 AWS IAM Identity Center를 추천합니다. 자세한 정보는 AWS IAM Identity Center 사용 설명서의 What is IAM Identity Center?를 참조하세요.

IAM 사용자 및 그룹

IAM 사용자는 단일 개인 또는 애플리케이션에 대한 특정 권한을 가진 ID입니다. 장기 자격 증명이 있는 IAM 사용자 대신 임시 자격 증명을 사용하는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 자격 증명 공급자와의 페더레이션을 사용하여 임시 자격 증명을 AWS 사용하여에 액세스하도록 인간 사용자에게 요구하기를 참조하세요.

IAM 그룹은 IAM 사용자 모음을 지정하고 대규모 사용자 집합에 대한 관리 권한을 더 쉽게 만듭니다. 자세한 내용은 IAM 사용 설명서의 IAM 사용자 사용 사례 섹션을 참조하세요.

IAM 역할

IAM 역할은 임시 자격 증명을 제공하는 특정 권한이 있는 자격 증명입니다. 사용자에서 IAM 역할(콘솔)로 전환하거나 또는 API 작업을 호출하여 역할을 수임할 수 있습니다. AWS CLI AWS 자세한 내용은 IAM 사용 설명서의 역할 수임 방법을 참조하세요.

IAM 역할은 페더레이션 사용자 액세스, 임시 IAM 사용자 권한, 교차 계정 액세스, 교차 서비스 액세스 및 Amazon EC2에서 실행되는 애플리케이션에 유용합니다. 자세한 내용은 IAM 사용 설명서의 교차 계정 리소스 액세스를 참조하세요.

정책을 사용하여 액세스 관리

정책을 AWS 생성하고 자격 증명 또는 리소스에 연결하여 AWS 에서 액세스를 제어합니다. 정책은 자격 증명 또는 리소스와 연결될 때 권한을 정의합니다.는 보안 주체가 요청할 때 이러한 정책을 AWS 평가합니다. 대부분의 정책은에 JSON 문서 AWS 로 저장됩니다. JSON 정책 문서에 대한 자세한 내용은 IAM 사용 설명서의 JSON 정책 개요 섹션을 참조하세요.

정책을 사용하여 관리자는 어떤 보안 주체가 어떤 리소스에 대해 어떤 조건에서 작업을 수행할 수 있는지 정의하여 누가 무엇을 액세스할 수 있는지 지정합니다.

기본적으로 사용자 및 역할에는 어떠한 권한도 없습니다. IAM 관리자는 IAM 정책을 생성하고 사용자가 수임할 수 있는 역할에 추가합니다. IAM 정책은 작업을 수행하기 위해 사용하는 방법과 관계없이 작업에 대한 권한을 정의합니다.

ID 기반 정책

ID 기반 정책은 ID(사용자, 사용자 그룹 또는 역할)에 연결하는 JSON 권한 정책 문서입니다. 이러한 정책은 자격 증명이 수행할 수 있는 작업, 대상 리소스 및 이에 관한 조건을 제어합니다. ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서에서 고객 관리형 정책으로 사용자 지정 IAM 권한 정의를 참조하세요.

ID 기반 정책은 인라인 정책(단일 ID에 직접 포함) 또는 관리형 정책(여러 ID에 연결된 독립 실행형 정책)일 수 있습니다. 관리형 정책 또는 인라인 정책을 선택하는 방법을 알아보려면 IAM 사용 설명서의 관리형 정책 및 인라인 정책 중에서 선택 섹션을 참조하세요.

리소스 기반 정책

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 예를 들어 IAM 역할 신뢰 정책 및 Amazon S3 버킷 정책이 있습니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다.

리소스 기반 정책은 해당 서비스에 있는 인라인 정책입니다. 리소스 기반 정책에서는 IAM의 AWS 관리형 정책을 사용할 수 없습니다.

기타 정책 유형

AWS 는 보다 일반적인 정책 유형에서 부여한 최대 권한을 설정할 수 있는 추가 정책 유형을 지원합니다.

- 권한 경계 - ID 기반 정책에서 IAM 엔터티에 부여할 수 있는 최대 권한을 설정합니다. 자세한 정보는 IAM 사용 설명서의 IAM 엔터티의 권한 범위를 참조하세요.
- 서비스 제어 정책(SCP) - AWS Organizations내 조직 또는 조직 단위에 대한 최대 권한을 지정합니다. 자세한 내용은AWS Organizations 사용 설명서의 서비스 제어 정책을 참조하세요.
- 리소스 제어 정책(RCP) – 계정의 리소스에 사용할 수 있는 최대 권한을 설정합니다. 자세한 내용은 AWS Organizations 사용 설명서의 리소스 제어 정책(RCP)을 참조하세요.
- 세션 정책 – 역할 또는 페더레이션 사용자에 대해 임시 세션을 프로그래밍 방식으로 생성할 때 파라미터로 전달하는 고급 정책입니다. 자세한 내용은 IAM 사용 설명서의 세션 정책을 참조하세요.

여러 정책 유형

여러 정책 유형이 요청에 적용되는 경우, 결과 권한은 이해하기가 더 복잡합니다. 에서 여러 정책 유형이 관련될 때 요청을 허용할지 여부를 AWS 결정하는 방법을 알아보려면 IAM 사용 설명서의 정책 평가 로직을 참조하세요.

AWS IoT TwinMaker 에서 IAM을 사용하는 방법

IAM을 사용하여에 대한 액세스를 관리하기 전에 사용할 수 있는 IAM 기능을 AWS IoT TwinMaker알아봅니다 AWS IoT TwinMaker.

에서 사용할 수 있는 IAM 기능 AWS IoT TwinMaker

IAM 특성	AWS IoT TwinMaker 지원
자격 증명 기반 정책	예
리소스 기반 정책	아니요
정책 작업	예
정책 리소스	예
정책 조건 키	예
ACL	아니요
ABAC(정책 내 태그)	부분적
임시 자격 증명	예
엔터티 권한	예
서비스 역할	예
서비스 연결 역할	아니요

AWS IoT TwinMaker 및 기타 AWS 서비스가 대부분의 IAM 기능과 작동하는 방법을 개괄적으로 알아보려면 AWS IAM Identity Center 사용 설명서의 AWS IAM으로 작업하는 서비스를 참조하세요.

에 대한 자격 증명 기반 정책 AWS IoT TwinMaker

ID 기반 정책 지원: 예

ID 기반 정책은 IAM 사용자, 사용자 그룹 또는 역할과 같은 ID에 연결할 수 있는 JSON 권한 정책 문서입니다. 이러한 정책은 사용자 및 역할이 어떤 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는지를 제어합니다. 자격 증명 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서에서 고객 관리형 정책으로 사용자 지정 IAM 권한 정의를 참조하세요.

IAM ID 기반 정책을 사용하면 허용되거나 거부되는 작업과 리소스뿐 아니라 작업이 허용되거나 거부되는 조건을 지정할 수 있습니다. JSON 정책에서 사용할 수 있는 모든 요소에 대해 알아보려면 IAM 사용 설명서의 IAM JSON 정책 요소 참조를 참조하세요.

에 대한 자격 증명 기반 정책 예제 AWS IoT TwinMaker

자격 AWS IoT TwinMaker 증명 기반 정책의 예를 보려면 섹션을 참조하세요에 대한 자격 증명 기반 정책 예제 AWS IoT TwinMaker.

내의 리소스 기반 정책 AWS IoT TwinMaker

리소스 기반 정책 지원: 아니요

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 리소스 기반 정책의 예제는 IAM 역할 신뢰 정책과 Amazon S3 버킷 정책입니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 정책이 연결된 리소스의 경우 정책은 지정된 보안 주체가 해당 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는지를 정의합니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다. 보안 주체에는 계정, 사용자, 역할, 페더레이션 사용자 또는이 포함될 수 있습니다 AWS 서비스.

교차 계정 액세스를 활성화하려는 경우, 전체 계정이나 다른 계정의 IAM 개체를 리소스 기반 정책의 보안 주체로 지정할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 IAM에서 교차 계정 리소스 액세스를 참조하세요.

에 대한 정책 작업 AWS IoT TwinMaker

정책 작업 지원: 예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

JSON 정책의 `Action`요소는 정책에서 액세스를 허용하거나 거부하는 데 사용할 수 있는 작업을 설명합니다. 연결된 작업을 수행할 수 있는 권한을 부여하기 위한 정책에 작업을 포함하세요.

AWS IoT TwinMaker 작업 목록을 보려면 서비스 승인 참조의에서 정의한 작업을 AWS IoT TwinMaker 참조하세요.

의 정책 작업은 작업 앞에 다음 접두사를 AWS IoT TwinMaker 사용합니다.

```
iottwinmaker
```

단일 문에서 여러 작업을 지정하려면 쉼표로 구분합니다.

```
"Action": [
   "iottwinmaker:action1",
   "iottwinmaker:action2"
         ]
```

자격 AWS IoT TwinMaker 증명 기반 정책의 예를 보려면 섹션을 참조하세요에 대한 자격 증명 기반 정책 예제 AWS IoT TwinMaker.

에 대한 정책 리소스 AWS IoT TwinMaker

정책 리소스 지원: 예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

`Resource` JSON 정책 요소는 작업이 적용되는 하나 이상의 객체를 지정합니다. 모범 사례에 따라 Amazon 리소스 이름(ARN)을 사용하여 리소스를 지정합니다. 리소스 수준 권한을 지원하지 않는 작업의 경우, 와일드카드(*)를 사용하여 해당 문이 모든 리소스에 적용됨을 나타냅니다.

```
"Resource": "*"
```

AWS IoT TwinMaker 리소스 유형 및 해당 ARNs 목록을 보려면 서비스 승인 참조의에서 정의한 리소스를 AWS IoT TwinMaker 참조하세요. 각 리소스의 ARN을 지정할 수 있는 작업을 알아보려면 AWS IoT TwinMaker에서 정의한 작업을 참조하세요.

자격 AWS IoT TwinMaker 증명 기반 정책의 예를 보려면 섹션을 참조하세요에 대한 자격 증명 기반 정책 예제 AWS IoT TwinMaker.

AWS IoT TwinMaker정책 조건 키

서비스별 정책 조건 키 지원: 예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

`Condition` 요소는 정의된 기준에 따라 문이 실행되는 시기를 지정합니다. 같음(equals) 또는 미만(less than)과 같은 조건 연산자를 사용하여 정책의 조건을 요청의 값과 일치시키는 조건식을 생성할 수

있습니다. 모든 AWS 전역 조건 키를 보려면 IAM 사용 설명서의 AWS 전역 조건 컨텍스트 키를 참조하세요.

AWS IoT TwinMaker 조건 키 목록을 보려면 서비스 승인 참조의에 대한 조건 키를 AWS IoT TwinMaker 참조하세요. 조건 키를 사용할 수 있는 작업과 리소스를 알아보려면 에서 정의한 작업을 AWS IoT TwinMaker 참조하세요.

자격 AWS IoT TwinMaker 증명 기반 정책의 예를 보려면 섹션을 참조하세요에 대한 자격 증명 기반 정책 예제 AWS IoT TwinMaker.

의 액세스 제어 목록(ACLs) AWS IoT TwinMaker

ACL 지원: 아니요

액세스 제어 목록(ACL)은 어떤 보안 주체(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는지를 제어합니다. ACL은 JSON 정책 문서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

를 사용한 ABAC(속성 기반 액세스 제어) AWS IoT TwinMaker

ABAC 지원(정책의 태그): 부분적

속성 기반 액세스 제어(ABAC)는 태그라고 불리는 속성을 기반으로 권한을 정의하는 권한 부여 전략입니다. IAM 엔터티 및 AWS 리소스에 태그를 연결한 다음 보안 주체의 태그가 리소스의 태그와 일치할 때 작업을 허용하는 ABAC 정책을 설계할 수 있습니다.

태그에 근거하여 액세스를 제어하려면 `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`또는 `aws:TagKeys`조건 키를 사용하여 정책의 조건 요소에 태그 정보를 제공합니다.

서비스가 모든 리소스 유형에 대해 세 가지 조건 키를 모두 지원하는 경우, 값은 서비스에 대해 예입니다. 서비스가 일부 리소스 유형에 대해서만 세 가지 조건 키를 모두 지원하는 경우, 값은 부분적입니다.

ABAC에 대한 자세한 내용은 IAM 사용 설명서의 ABAC 권한 부여를 통한 권한 정의를 참조하세요. ABAC 설정 단계가 포함된 자습서를 보려면 IAM 사용 설명서의 속성 기반 액세스 제어(ABAC) 사용을 참조하세요.

에서 임시 자격 증명 사용 AWS IoT TwinMaker

임시 자격 증명 지원: 예

임시 자격 증명은 AWS 리소스에 대한 단기 액세스를 제공하며 페더레이션 또는 전환 역할을 사용할 때 자동으로 생성됩니다. 장기 액세스 키를 사용하는 대신 임시 자격 증명을 동적으로 생성하는 것이

AWS 좋습니다. 자세한 내용은 IAM 사용 설명서의 IAM의 임시 보안 자격 증명 및 IAM으로 작업하는 AWS 서비스 섹션을 참조하세요.

에 대한 교차 서비스 보안 주체 권한 AWS IoT TwinMaker

전달 액세스 세션(FAS) 지원: 예

전달 액세스 세션(FAS)은를 호출하는 보안 주체의 권한을 다운스트림 서비스에 AWS 서비스 대한 요청과 AWS 서비스함께 사용합니다. FAS 요청 시 정책 세부 정보는 전달 액세스 세션을 참조하세요.

에 대한 서비스 역할 AWS IoT TwinMaker

서비스 역할 지원: 예

서비스 역할은 서비스가 사용자를 대신하여 작업을 수행하는 것으로 가정하는 IAM 역할입니다. IAM 관리자는 IAM 내에서 서비스 역할을 생성, 수정 및 삭제할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 AWS 서비스 AWS에 권한을 위임할 역할 생성을 참조하세요.

> **Warning**
>
> 서비스 역할에 대한 권한을 변경하면 AWS IoT TwinMaker 기능이 중단될 수 있습니다. 에서 관련 지침을 AWS IoT TwinMaker 제공하는 경우에만 서비스 역할을 편집합니다.

에 대한 서비스 연결 역할 AWS IoT TwinMaker

서비스 연결 역할 지원: 아니요

서비스 연결 역할은에 연결된 서비스 역할의 한 유형입니다 AWS 서비스. 서비스는 사용자를 대신하여 태스크를 수행하기 위해 역할을 수임할 수 있습니다. 서비스 연결 역할은에 나타나 AWS 계정 며 서비스가 소유합니다. IAM 관리자는 서비스 연결 역할의 권한을 볼 수 있지만 편집은 할 수 없습니다.

서비스 연결 역할 생성 또는 관리에 대한 자세한 내용은 IAM으로 작업하는AWS 서비스를 참조하세요. 서비스 연결 역할 열에서 Yes가 포함된 서비스를 테이블에서 찾습니다. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 예(Yes) 링크를 선택합니다.

에 대한 자격 증명 기반 정책 예제 AWS IoT TwinMaker

기본적으로 사용자 및 역할에는 AWS IoT TwinMaker 리소스를 생성하거나 수정할 수 있는 권한이 없습니다. 사용자에게 사용자가 필요한 리소스에서 작업을 수행할 권한을 부여하려면 IAM 관리자가 IAM 정책을 생성하면 됩니다.

이러한 예제 JSON 정책 문서를 사용하여 IAM ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 IAM 정책 생성(콘솔)을 참조하세요.

각 리소스 유형에 대한 ARNs 형식을 포함하여 AWS IoT TwinMaker에서 정의한 작업 및 리소스 유형에 대한 자세한 내용은 서비스 승인 참조의 에 대한 작업, 리소스 및 조건 키를 AWS IoT TwinMaker 참조하세요.

주제

- 정책 모범 사례
- AWS IoT TwinMaker 콘솔 사용
- 사용자가 자신의 고유한 권한을 볼 수 있도록 허용

정책 모범 사례

자격 증명 기반 정책에 따라 계정에서 사용자가 AWS IoT TwinMaker 리소스를 생성, 액세스 또는 삭제할 수 있는지 여부가 결정됩니다. 이 작업으로 인해 AWS 계정에 비용이 발생할 수 있습니다. ID 기반 정책을 생성하거나 편집할 때는 다음 지침과 권장 사항을 따르세요.

- AWS 관리형 정책을 시작하고 최소 권한으로 전환 - 사용자 및 워크로드에 권한 부여를 시작하려면 많은 일반적인 사용 사례에 대한 권한을 부여하는 AWS 관리형 정책을 사용합니다. 에서 사용할 수 있습니다 AWS 계정. 사용 사례에 맞는 AWS 고객 관리형 정책을 정의하여 권한을 추가로 줄이는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 AWS 관리형 정책 또는 AWS 직무에 대한 관리형 정책을 참조하세요.
- 최소 권한 적용 – IAM 정책을 사용하여 권한을 설정하는 경우, 작업을 수행하는 데 필요한 권한만 부여합니다. 이렇게 하려면 최소 권한으로 알려진 특정 조건에서 특정 리소스에 대해 수행할 수 있는 작업을 정의합니다. IAM을 사용하여 권한을 적용하는 방법에 대한 자세한 정보는 IAM 사용 설명서에 있는 IAM의 정책 및 권한을 참조하세요.
- IAM 정책의 조건을 사용하여 액세스 추가 제한 – 정책에 조건을 추가하여 작업 및 리소스에 대한 액세스를 제한할 수 있습니다. 예를 들어, SSL을 사용하여 모든 요청을 전송해야 한다고 지정하는 정책 조건을 작성할 수 있습니다. AWS 서비스와 같은 특정를 통해 사용되는 경우 조건을 사용하여 서비스 작업에 대한 액세스 권한을 부여할 수도 있습니다 CloudFormation. 자세한 내용은 IAM 사용 설명서의 IAM JSON 정책 요소: 조건을 참조하세요.
- IAM Access Analyzer를 통해 IAM 정책을 확인하여 안전하고 기능적인 권한 보장 - IAM Access Analyzer에서는 IAM 정책 언어(JSON)와 모범 사례가 정책에서 준수되도록 새로운 및 기존 정책을 확인합니다. IAM Access Analyzer는 100개 이상의 정책 확인 항목과 실행 가능한 추천을 제공하

여 안전하고 기능적인 정책을 작성하도록 돕습니다. 자세한 내용은 IAM 사용 설명서의 IAM Access Analyzer에서 정책 검증을 참조하세요.

- 다중 인증(MFA) 필요 -에서 IAM 사용자 또는 루트 사용자가 필요한 시나리오가 있는 경우 추가 보안을 위해 MFA를 AWS 계정합니다. API 작업을 직접적으로 호출할 때 MFA가 필요하면 정책에 MFA 조건을 추가합니다. 자세한 내용은 IAM 사용 설명서의 MFA를 통한 보안 API 액세스를 참조하세요.

IAM의 모범 사례에 대한 자세한 내용은 IAM 사용 설명서의 IAM의 보안 모범 사례를 참조하세요.

AWS IoT TwinMaker 콘솔 사용

AWS IoT TwinMaker 콘솔에 액세스하려면 최소 권한 집합이 있어야 합니다. 이러한 권한은의 AWS IoT TwinMaker 리소스에 대한 세부 정보를 나열하고 볼 수 있도록 허용해야 합니다 AWS 계정. 최소 필수 권한보다 더 제한적인 ID 기반 정책을 생성하는 경우, 콘솔이 해당 정책에 연결된 엔티티(사용자 또는 역할)에 대해 의도대로 작동하지 않습니다.

AWS CLI 또는 AWS API만 호출하는 사용자에게는 최소 콘솔 권한을 허용할 필요가 없습니다. 대신, 수행하려는 API 작업과 일치하는 작업에만 액세스할 수 있도록 합니다.

사용자와 역할이 AWS IoT TwinMaker 콘솔을 계속 사용할 수 있도록 하려면 AWS IoT TwinMaker `ConsoleAccess` 또는 `ReadOnly` AWS 관리형 정책도 엔터티에 연결합니다. 자세한 내용은 AWS IAM Identity Center 사용 설명서의 사용자에게 권한 추가를 참조하십시오.

사용자가 자신의 고유한 권한을 볼 수 있도록 허용

이 예제는 IAM 사용자가 자신의 사용자 ID에 연결된 인라인 및 관리형 정책을 볼 수 있도록 허용하는 정책을 생성하는 방법을 보여줍니다. 이 정책에는 콘솔에서 또는 AWS CLI 또는 AWS API를 사용하여 프로그래밍 방식으로이 작업을 완료할 수 있는 권한이 포함됩니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
```

```
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS IoT TwinMaker 자격 증명 및 액세스 문제 해결

다음 정보를 사용하여 및 IAM으로 작업할 때 발생할 수 있는 일반적인 문제를 진단 AWS IoT TwinMaker 하고 수정할 수 있습니다.

주제

- 에서 작업을 수행할 권한이 없음 AWS IoT TwinMaker
- iam:PassRole을 수행하도록 인증되지 않음
- 내 외부의 사람이 내 AWS IoT TwinMaker 리소스에 액세스 AWS 계정 하도록 허용하고 싶습니다.

에서 작업을 수행할 권한이 없음 AWS IoT TwinMaker

작업을 수행할 권한이 없다는 오류가 표시되면 작업을 수행할 수 있도록 정책을 업데이트해야 합니다.

다음의 예제 오류는 `mateojackson` IAM 사용자가 콘솔을 사용하여 가상 *`my-example-widget`* 리소스에 대한 세부 정보를 보려고 하지만 가상 `iottwinmaker:`*`GetWidget`* 권한이 없을 때 발생합니다.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 iottwinmaker:GetWidget on resource: my-example-widget
```

이 경우, `iottwinmaker:GetWidget` 작업을 사용하여 *my-example-widget* 리소스에 액세스할 수 있도록 `mateojackson` 사용자 정책을 업데이트해야 합니다.

도움이 필요한 경우 AWS 관리자에게 문의하세요. 관리자는 로그인 자격 증명을 제공한 사람입니다.

iam:PassRole을 수행하도록 인증되지 않음

`iam:PassRole` 작업을 수행할 수 있는 권한이 없다는 오류가 수신되면 AWS IoT TwinMaker에 역할을 전달할 수 있도록 정책을 업데이트해야 합니다.

일부 AWS 서비스 에서는 새 서비스 역할 또는 서비스 연결 역할을 생성하는 대신 기존 역할을 해당 서비스에 전달할 수 있습니다. 이렇게 하려면 역할을 서비스에 전달할 권한이 있어야 합니다.

다음 예 오류는 `marymajor`라는 IAM 사용자가 콘솔을 사용하여 AWS IoT TwinMaker에서 작업을 수행하려고 하는 경우에 발생합니다. 하지만 작업을 수행하려면 서비스 역할이 부여한 권한이 서비스에 있어야 합니다. Mary는 서비스에 역할을 전달할 권한이 없습니다.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

이 경우, Mary가 `iam:PassRole`작업을 수행할 수 있도록 Mary의 정책을 업데이트해야 합니다.

도움이 필요한 경우 AWS 관리자에게 문의하세요. 관리자는 로그인 자격 증명을 제공한 사람입니다.

내 외부의 사람이 내 AWS IoT TwinMaker 리소스에 액세스 AWS 계정 하도록 허용하고 싶습니다.

다른 계정의 사용자 또는 조직 외부의 사람이 리소스에 액세스할 때 사용할 수 있는 역할을 생성할 수 있습니다. 역할을 수임할 신뢰할 수 있는 사람을 지정할 수 있습니다. 리소스 기반 정책 또는 액세스 제어 목록(ACL)을 지원하는 서비스의 경우, 이러한 정책을 사용하여 다른 사람에게 리소스에 대한 액세스 권한을 부여할 수 있습니다.

자세한 내용은 다음을 참조하세요.

- 에서 이러한 기능을 AWS IoT TwinMaker 지원하는지 여부를 알아보려면 섹션을 참조하세요AWS IoT TwinMaker 에서 IAM을 사용하는 방법.

- 소유 AWS 계정 한의 리소스에 대한 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 소유한 다른의 IAM 사용자에게 액세스 권한 제공을 참조 AWS 계정 하세요.
- 리소스에 대한 액세스 권한을 타사에 제공하는 방법을 알아보려면 IAM 사용 설명서의 타사 AWS 계정 소유에 대한 액세스 권한 제공을 AWS 계정참조하세요.
- ID 페더레이션을 통해 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 외부에서 인증된 사용자에게 액세스 권한 제공(ID 페더레이션)을 참조하세요.
- 크로스 계정 액세스에 대한 역할과 리소스 기반 정책 사용의 차이점을 알아보려면 IAM 사용 설명서의 IAM의 크로스 계정 리소스 액세스를 참조하세요.

에 서비스 연결 역할 사용 AWS IoT TwinMaker

AWS IoT TwinMaker 는 AWS Identity and Access Management (IAM) 서비스 연결 역할을 사용합니다. 서비스 연결 역할은 직접 연결된 고유한 유형의 IAM 역할입니다 AWS IoT TwinMaker. 서비스 연결 역할은에서 사전 정의 AWS IoT TwinMaker 하며 서비스가 사용자를 대신하여 다른 AWS 서비스를 호출하는 데 필요한 모든 권한을 포함합니다.

필요한 권한을 수동으로 추가할 필요가 없으므로 서비스 연결 역할을 더 AWS IoT TwinMaker 쉽게 설정할 수 있습니다.는 서비스 연결 역할의 권한을 AWS IoT TwinMaker 정의하며, 달리 정의되지 않은 한 만 해당 역할을 수임 AWS IoT TwinMaker 할 수 있습니다. 정의된 권한에는 신뢰 정책과 권한 정책이 포함되며 이 권한 정책은 다른 IAM 엔터티에 연결할 수 없습니다.

먼저 관련 리소스를 삭제한 후에만 서비스 연결 역할을 삭제할 수 있습니다. 이렇게 하면 AWS IoT TwinMaker 리소스에 대한 액세스 권한을 실수로 제거할 수 없기 때문에 리소스가 보호됩니다.

서비스 연결 역할을 지원하는 다른 서비스에 대한 자세한 내용은 AWS IAM으로 작업하는 서비스를 참조하고 서비스 연결 역할 열에서 예인 서비스를 찾습니다. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 예 링크를 선택합니다.

에 대한 서비스 연결 역할 권한 AWS IoT TwinMaker

AWS IoT TwinMaker 는 AWSServiceRoleForIoTTwinMaker라는 서비스 연결 역할을 사용합니다. AWS IoT TwinMaker 는 다른 AWS 서비스를 호출하고 사용자를 대신하여 리소스를 동기화할 수 있습니다.

AWSServiceRoleForIoTTwinMaker 서비스 연결 역할은 역할을 수임하기 위해 다음 서비스를 신뢰합니다.

- `iottwinmaker.amazonaws.com`

AWSIoTTwinMakerServiceRolePolicy라는 역할 권한 정책은가 지정된 리소스에서 다음 작업을 완료 AWS IoT TwinMaker 하도록 허용합니다.

- 작업: `all your iotsitewise asset and asset-model resources`에 대한 `iotsitewise:DescribeAsset, iotsitewise:ListAssets, iotsitewise:DescribeAssetModel, and iotsitewise:ListAssetModels, iottwinmaker:GetEntity, iottwinmaker:CreateEntity, iottwinmaker:UpdateEntity, iottwinmaker:DeleteEntity, iottwinmaker:ListEntities, iottwinmaker:GetComponentType, iottwinmaker:CreateComponentType, iottwinmaker:UpdateComponentType, iottwinmaker:DeleteComponentType, iottwinmaker:ListComponentTypes`

사용자, 그룹 또는 역할이 서비스 연결 역할을 생성, 편집 또는 삭제할 수 있도록 사용 권한을 구성해야 합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 권한을 참조하세요.

에 대한 서비스 연결 역할 생성 AWS IoT TwinMaker

서비스 연결 역할은 수동으로 생성할 필요가 없습니다. 에서 AWS IoT SiteWise 자산과 자산 모델을 동기화하면(자산 동기화) AWS CLI또는 AWS API AWS Management Console가 서비스 연결 역할을 AWS IoT TwinMaker 생성합니다.

이 서비스 연결 역할을 삭제했다가 다시 생성해야 하는 경우 동일한 프로세스를 사용하여 계정에서 역할을 다시 생성할 수 있습니다. AWS IoT SiteWise 자산과 자산 모델을 동기화하면(자산 동기화)가 서비스 연결 역할을 다시 AWS IoT TwinMaker 생성합니다.

IAM 콘솔을 사용하여 "IoT TwinMaker - 관리형 역할" 사용 사례로 서비스 연결 역할을 생성할 수도 있습니다. AWS CLI 또는 AWS API에서 서비스 이름을 사용하여 `iottwinmaker.amazonaws.com` 서비스 연결 역할을 생성합니다. 자세한 내용은 IAM 사용자 설명서의 서비스 연결 역할 생성 섹션을 참조하세요. 이 서비스 연결 역할을 삭제하면 동일한 프로세스를 사용하여 역할을 다시 생성할 수 있습니다.

에 대한 서비스 연결 역할 편집 AWS IoT TwinMaker

AWS IoT TwinMaker 에서는 AWSServiceRoleForIoTTwinMaker 서비스 연결 역할을 편집할 수 없습니다. 서비스 연결 역할을 생성한 후에는 다양한 개체가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 편집을 참조하세요.

에 대한 서비스 연결 역할 삭제 AWS IoT TwinMaker

서비스 연결 역할이 필요한 기능 또는 서비스가 더 이상 필요 없는 경우에는 해당 역할을 삭제하는 것이 좋습니다. 따라서 적극적으로 모니터링하거나 유지하지 않는 미사용 엔터티가 없도록 합니다. 그러나 역할을 수동으로 삭제하려면 먼저 서비스 연결 역할을 아직 사용 중인 모든 serviceLinked-workspace를 정리해야 합니다.

> **Note**
>
> 리소스를 삭제하려고 할 때 AWS IoT TwinMaker 서비스가 역할을 사용하는 경우 삭제에 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하세요.

IAM을 사용하여 수동으로 서비스 연결 역할을 삭제하려면 다음을 수행하세요.

IAM 콘솔 AWS CLI, 또는 AWS API를 사용하여 AWSServiceRoleForIoTTwinMaker 서비스 연결 역할을 삭제합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 삭제를 참조하십시오.

AWS IoT TwinMaker 서비스 연결 역할에 지원되는 리전

AWS IoT TwinMaker 는 서비스를 사용할 수 있는 모든 리전에서 서비스 연결 역할 사용을 지원합니다. 자세한 내용은 AWS 리전 및 엔드포인트 섹션을 참조하세요.

AWS 에 대한 관리형 정책 AWS IoT TwinMaker

사용자, 그룹 및 역할에 권한을 추가하려면 직접 정책을 작성하는 것보다 AWS 관리형 정책을 사용하는 것이 더 쉽습니다. 팀에 필요한 권한만 제공하는 IAM 고객 관리형 정책을 생성하기 위해서는 시간과 전문 지식이 필요합니다. 빠르게 시작하려면 AWS 관리형 정책을 사용할 수 있습니다. 이 정책은 일반적인 사용 사례를 다루며 사용자의 AWS 계정에서 사용할 수 있습니다. AWS 관리형 정책에 대한 자세한 내용은 IAM 사용 설명서의 AWS 관리형 정책을 참조하세요.

AWS 서비스는 AWS 관리형 정책을 유지 관리하고 업데이트합니다. AWS 관리형 정책에서는 권한을 변경할 수 없습니다. 서비스에서 때때로 추가 권한을 AWS 관리형 정책에 추가하여 새로운 기능을 지원합니다. 이 유형의 업데이트는 정책이 연결된 모든 ID(사용자, 그룹 및 역할)에 적용됩니다. 서비스는 새로운 기능이 시작되거나 새 작업을 사용할 수 있을 때 AWS 관리형 정책에 업데이트됩니다. 서비스는 AWS 관리형 정책에서 권한을 제거하지 않으므로 정책 업데이트로 인해 기존 권한이 손상되지 않습니다.

또한는 여러 서비스에 걸쳐 있는 직무에 대한 관리형 정책을 AWS 지원합니다. 예를 들어 ReadOnlyAccess AWS 관리형 정책은 모든 AWS 서비스 및 리소스에 대한 읽기 전용 액세스를 제공합니다. 서비스가 새 기능을 시작하면는 새 작업 및 리소스에 대한 읽기 전용 권한을 AWS 추가합니다. 직무 정책의 목록과 설명은 IAM 사용 설명서의 직무에 관한AWS 관리형 정책을 참조하세요.

AWS 관리형 정책: AWSIoTTwinMakerServiceRolePolicy

AWSIoTTwinMakerServiceRolePolicy를 IAM 엔터티에 연결할 수 없습니다. 이 정책은 이(가) 사용자를 대신하여 작업을 수행할 수 있도록 서비스 연결 역할에 연결됩니다. 자세한 내용은 에 대한 서비스 연결 역할 권한 AWS IoT TwinMaker 단원을 참조하십시오.

AWSIoTTwinMakerServiceRolePolicy라는 역할 권한 정책은가 지정된 리소스에서 다음 작업을 완료 AWS IoT TwinMaker 하도록 허용합니다.

- 작업: `all your iotsitewise asset and asset-model resources`에 대한 `iotsitewise:DescribeAsset, iotsitewise:ListAssets, iotsitewise:DescribeAssetModel, and iotsitewise:ListAssetModels, iottwinmaker:GetEntity, iottwinmaker:CreateEntity, iottwinmaker:UpdateEntity, iottwinmaker:DeleteEntity, iottwinmaker:ListEntities, iottwinmaker:GetComponentType, iottwinmaker:CreateComponentType, iottwinmaker:UpdateComponentType, iottwinmaker:DeleteComponentType, iottwinmaker:ListComponentTypes`

JSON

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

```
                "iotsitewise:DescribeAsset"
            ],
            "Resource": [
                "arn:aws:iotsitewise:*:*:asset/*"
            ]
        },
        {
            "Sid": "SiteWiseAssetModelReadAccess",
            "Effect": "Allow",
            "Action": [
                "iotsitewise:DescribeAssetModel"
            ],
            "Resource": [
                "arn:aws:iotsitewise:*:*:asset-model/*"
            ]
        },
        {
            "Sid": "SiteWiseAssetModelAndAssetListAccess",
            "Effect": "Allow",
            "Action": [
                "iotsitewise:ListAssets",
                "iotsitewise:ListAssetModels"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Sid": "TwinMakerAccess",
            "Effect": "Allow",
            "Action": [
                "iottwinmaker:GetEntity",
                "iottwinmaker:CreateEntity",
                "iottwinmaker:UpdateEntity",
                "iottwinmaker:DeleteEntity",
                "iottwinmaker:ListEntities",
                "iottwinmaker:GetComponentType",
                "iottwinmaker:CreateComponentType",
                "iottwinmaker:UpdateComponentType",
                "iottwinmaker:DeleteComponentType",
                "iottwinmaker:ListComponentTypes"
            ],
            "Resource": [
                "arn:aws:iottwinmaker:*:*:workspace/*"
```

```
            ],
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "iottwinmaker:linkedServices": [
                        "IOTSITEWISE"
                    ]
                }
            }
        }
    ]
}
```

AWS IoT TwinMaker AWS 관리형 정책에 대한 업데이트

이 서비스가 이러한 변경 사항을 추적하기 시작한 이후부터의 AWS 관리형 정책 업데이트에 대한 세부 정보를 봅니다. 이 페이지의 변경 사항에 대한 자동 알림을 받으려면 문서 기록 페이지에서 RSS 피드를 구독하세요.

변경	설명	Date
AWSIoTTwinMakerServiceRolePolicy – 정책 추가	AWS IoT TwinMaker 는가 지정된 리소스에서 다음 작업을 완료 AWS IoT TwinMaker 하도록 허용하는 AWSIoTTwinMakerServiceRolePolicy라는 역할 권한 정책을 추가했습니다. • 작업: `all your iotsitewise asset and asset-model resources` 에 대한 `iotsitewise:DescribeAsset, iotsitewise:ListAssets, iotsitewise:Descri`	2023년 11월 17일

변경	설명	Date
	beAssetModel, and iotsitewise:ListAssetModels, iottwinmaker:GetEntity, iottwinmaker:CreateEntity, iottwinmaker:UpdateEntity, iottwinmaker:DeleteEntity, iottwinmaker:ListEntities, iottwinmaker:GetComponentType, iottwinmaker:CreateComponentType, iottwinmaker:UpdateComponentType, iottwinmaker:DeleteComponentType, iottwinmaker:ListComponentTypes 자세한 내용은 에 대한 서비스 연결 역할 권한 AWS IoT TwinMaker 단원을 참조하십시오.	
에서 변경 내용 추적 시작	가 AWS 관리형 정책에 대한 변경 내용 추적을 시작했습니다.	2022년 5월 11일

AWS IoT TwinMaker 및 인터페이스 VPC 엔드포인트(AWS PrivateLink)

인터페이스 VPC 엔드포인트를 생성하여 Virtual Private Cloud(VPC)와 AWS IoT TwinMaker 간에 프라이빗 연결을 설정할 수 있습니다. 인터페이스 엔드포인트는 인터넷 게이트웨이AWS PrivateLink, NAT(네트워크 주소 변환) 디바이스, VPN 연결 또는 AWS Direct Connect 연결 없이 비공개로 AWS IoT TwinMaker APIs에 액세스하는 데 사용할 수 있는 로 구동됩니다.는 인터페이스 엔드포인트를 통해 IPv4 및 IPv6(듀얼 스택)를 모두 AWS IoT TwinMaker 지원합니다. VPC의 인스턴스는 AWS IoT TwinMaker APIs. VPC와 간의 트래픽 AWS IoT TwinMaker 은 Amazon 네트워크를 벗어나지 않습니다.

각 인터페이스 엔드포인트는 서브넷에서 하나 이상의 Elastic Network Interfaces로 표현됩니다.

자세한 내용은 Amazon VPC 사용 설명서의 인터페이스 VPC 엔드포인트(AWS PrivateLink)를 참조하세요.

AWS IoT TwinMaker VPC 엔드포인트에 대한 고려 사항

에 대한 인터페이스 VPC 엔드포인트를 설정하기 전에 Amazon VPC 사용 설명서의 인터페이스 엔드포인트 속성 및 제한 사항을 AWS IoT TwinMaker검토하세요.

AWS IoT TwinMaker 는 VPC에서 모든 API 작업을 호출할 수 있도록 지원합니다.

- 데이터 영역 API 작업의 경우. 다음 엔드포인트를 사용하십시오.

  ```
  data.iottwinmaker.region.amazonaws.com
  ```

  데이터 영역 API 작업에는 다음이 포함됩니다.

  - GetPropertyValue
  - GetPropertyValueHistory
  - BatchPutPropertyValues
- 컨트롤 플레인 API 작업의 경우, 다음 엔드포인트를 사용하십시오.

  ```
  api.iottwinmaker.region.amazonaws.com
  ```

  지원되는 컨트롤 플레인 API 작업에는 다음이 포함됩니다.

  - CreateComponentType

- CreateEntity
- CreateScene
- CreateWorkspace
- DeleteComponentType
- DeleteEntity
- DeleteScene
- DeleteWorkspace
- GetComponentType
- GetEntity
- GetScene
- GetWorkspace
- ListComponentTypes
- ListComponentTypes
- ListEntities
- ListScenes
- ListTagsForResource
- ListWorkspaces
- TagResource
- UntagResource
- UpdateComponentType
- UpdateEntity
- UpdateScene
- UpdateWorkspace

에 대한 인터페이스 VPC 엔드포인트 생성 AWS IoT TwinMaker

Amazon VPC 콘솔 또는 AWS Command Line Interface ()를 사용하여 AWS IoT TwinMaker 서비스에 대한 VPC 엔드포인트를 생성할 수 있습니다AWS CLI. 자세한 내용은 Amazon VPC 사용 설명서의 인터페이스 엔드포인트 생성을 참조하세요.

다음 서비스 이름을 AWS IoT TwinMaker 사용하는에 대한 VPC 엔드포인트를 생성합니다.

- 데이터 영역 API 작업의 경우 다음 서비스 이름을 사용하십시오.

```
com.amazonaws.region.iottwinmaker.data
```

- 컨트롤 플레인 API 작업의 경우 다음 서비스 이름을 사용합니다.

```
com.amazonaws.region.iottwinmaker.api
```

엔드포인트에 대해 프라이빗 DNS를 활성화하는 경우 리전의 기본 DNS 이름, 예를 들어 AWS IoT TwinMaker 를 사용하여에 API 요청을 할 수 있습니다`iottwinmaker.us-east-1.amazonaws.com`.

자세한 내용은 Amazon VPC 사용 설명서의 인터페이스 엔드포인트를 통해 서비스 액세스를 참조하세요.

AWS IoT TwinMaker PrivateLink는 다음 리전에서 지원됩니다.

- us-east-1

  ControlPlane 서비스는 다음과 같은 가용 영역에서 지원됩니다: `use1-az1`, `use1-az2` 및 `use1-az6`.

  DataPlane 서비스는 다음과 같은 가용 영역에서 지원됩니다:`use1-az1`, `use1-az2` 및 `use1-az4`.

- us-west-2

  ControlPlane 및 DataPlane 서비스는 `usw2-az1`, `usw2-az2` 및 `usw2-az3` 가용 영역에서 지원됩니다.

- eu-west-1
- eu-central-1
- ap-southeast-1
- ap-southeast-2

가용 영역에 대한 자세한 내용은 AWS 리소스의 가용 영역 IDs - AWS 리소스 액세스 관리자를 참조하세요.

인터페이스 VPC 엔드포인트를 AWS IoT TwinMaker 통해 액세스

인터페이스 엔드포인트를 생성할 때는 통신에 사용할 수 있는 엔드포인트별 DNS 호스트 이름을 AWS IoT TwinMaker 생성합니다 AWS IoT TwinMaker. 프라이빗 DNS 옵션은 기본적으로 활성화되어 있습니다. 자세한 내용은 Amazon VPC 사용 설명서의 프라이빗 호스팅 영역 사용을 참조하세요.

엔드포인트에 프라이빗 DNS를 활성화하면 다음 VPC 엔드포인트 중 하나를 통해 AWS IoT TwinMaker 에 API 요청을 보낼 수 있습니다.

- 데이터 영역 API 작업의 경우, 다음 엔드포인트를 사용하십시오. *##*을 사용자의 AWS 리전으로 바꾸십시오.

```
data.iottwinmaker.region.amazonaws.com
```

- 컨트롤 플레인 API 작업의 경우, 다음 엔드포인트를 사용하십시오. *##*을 사용자의 AWS 리전으로 바꾸십시오.

```
api.iottwinmaker.region.amazonaws.com
```

엔드포인트에 대한 프라이빗 DNS를 비활성화한 경우, 엔드포인트를 통해 AWS IoT TwinMaker 에 액세스하려면 다음을 수행해야 합니다.

- API 요청에서 VPC 엔드포인트 URL을 지정합니다.
  - 데이터 영역 API 작업의 경우 다음 엔드포인트 URL을 사용합니다. *vpc-endpoint-id*와 *##*을 VPC 엔드포인트 ID 및 리전으로 바꾸세요.

```
vpc-endpoint-id.data.iottwinmaker.region.vpce.amazonaws.com
```

  - 컨트롤 플레인 API 작업의 경우, 다음 엔드포인트 URL을 사용하십시오. *vpc-endpoint-id*와 *##*을 VPC 엔드포인트 ID 및 리전으로 바꾸세요.

```
vpc-endpoint-id.api.iottwinmaker.region.vpce.amazonaws.com
```

- 호스트 접두사 삽입을 비활성화합니다. AWS CLI 및 AWS SDKs는 각 API 작업을 호출할 때 서비스 엔드포인트 앞에 다양한 호스트 접두사를 추가합니다. 이로 인해 VPC 엔드포인트를 지정할 AWS IoT TwinMaker 때 AWS CLI 및 AWS SDKs에 대해 잘못된 URLs을 생성합니다.

> **Important**
>
> AWS CLI 또는 AWS Tools for PowerShell에서 호스트 접두사 삽입을 비활성화할 수 없습니다. 즉, 프라이빗 DNS를 비활성화한 경우 또는를 사용하여 AWS CLI VPC 엔드포인트를 AWS IoT TwinMaker 통해 AWS Tools for PowerShell 에 액세스할 수 없습니다. 이러한 도구를 사용하여 엔드포인트를 AWS IoT TwinMaker 통해에 액세스하려면 프라이빗 DNS를 활성화합니다.

AWS SDK에서 호스트 접두사 삽입을 비활성화하는 방법에 대한 자세한 내용은 각 SDK에 대한 다음 설명서 섹션을 참조하세요.

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Go v2
- AWS SDK for Java
- AWS SDK for Java 2.x
- AWS SDK for JavaScript
- AWS SDK for .NET
- AWS SDK for PHP
- AWS SDK for Python (Boto3)
- AWS SDK for Ruby

자세한 내용은 Amazon VPC 사용 설명서의 인터페이스 엔드포인트를 통해 서비스 액세스를 참조하세요.

에 대한 VPC 엔드포인트 정책 생성 AWS IoT TwinMaker

AWS IoT TwinMaker에 대한 액세스를 제어하는 VPC 엔드포인트에 엔드포인트 정책을 연결할 수 있습니다. 이 정책은 다음 정보를 지정합니다.

- 작업을 수행할 수 있는 위탁자.
- 수행할 수 있는 작업.
- 작업을 수행할 수 있는 리소스.

자세한 내용은 Amazon VPC 사용 설명서의 VPC 엔드포인트를 통해 서비스에 대한 액세스 제어를 참조하세요.

예: AWS IoT TwinMaker 작업에 대한 VPC 엔드포인트 정책

다음은에 대한 엔드포인트 정책의 예입니다 AWS IoT TwinMaker. 엔드포인트에 연결되면이 정책은 `123456789012` 모든 리소스의 AWS 계정에 `iottwinmakeradmin` 있는 IAM 사용자에게 나열된 AWS IoT TwinMaker 작업에 대한 액세스 권한을 부여합니다.

```
{
   "Statement":[
      {
        "Principal": {
            "AWS": "arn:aws:iam::123456789012:user/role"
                },
         "Resource": "*",
         "Effect":"Allow",
         "Action":[
            "iottwinmaker:CreateEntity",
            "iottwinmaker:GetScene",
            "iottwinmaker:ListEntities"
         ]
        }
    ]
}
```

에 대한 규정 준수 검증 AWS IoT TwinMaker

AWS 서비스 가 특정 규정 준수 프로그램의 범위 내에 있는지 알아보려면 AWS 서비스 규정 준수 프로그램 범위 내 참조하고 관심 있는 규정 준수 프로그램을 선택합니다. 일반 정보는 AWS 규정 준수 프로그램.

를 사용하여 타사 감사 보고서를 다운로드할 수 있습니다 AWS Artifact. 자세한 내용은 Downloading Reports inDownloading AWS Artifact을 참조하세요.

사용 시 규정 준수 책임은 데이터의 민감도, 회사의 규정 준수 목표 및 관련 법률과 규정에 따라 AWS 서비스 결정됩니다. 사용 시 규정 준수 책임에 대한 자세한 내용은 AWS 보안 설명서를 AWS 서비스참조하세요.

의 복원력 AWS IoT TwinMaker

AWS 글로벌 인프라는 AWS 리전 및 가용 영역을 중심으로 구축됩니다.는 물리적으로 분리되고 격리된 여러 가용 영역을 AWS 리전 제공하며,이 가용 영역은 지연 시간이 짧고 처리량이 높으며 중복성이 높은 네트워킹과 연결됩니다. 가용 영역을 사용하면 중단 없이 영역 간에 자동으로 장애 극복 조치가 이루어지는 애플리케이션 및 데이터베이스를 설계하고 운영할 수 있습니다. 가용 영역은 기존의 단일 또는 다중 데이터 센터 인프라보다 가용성, 내결함성, 확장성이 뛰어납니다.

AWS 리전 및 가용 영역에 대한 자세한 내용은 AWS 글로벌 인프라를 참조하세요.

AWS 글로벌 인프라 외에도는 데이터 복원력 및 백업 요구 사항을 지원하는 몇 가지 기능을 AWS IoT TwinMaker 제공합니다.

의 인프라 보안 AWS IoT TwinMaker

관리형 서비스인는 Amazon Web Services: 보안 프로세스 개요 백서에 설명된 AWS 글로벌 네트워크 보안 절차로 AWS IoT TwinMaker 보호됩니다.

AWS 에서 게시한 API 호출을 사용하여 네트워크를 AWS IoT TwinMaker 통해에 액세스합니다. 클라이언트가 전송 계층 보안(TLS) 1.2 이상을 지원해야 합니다. TLS 1.3 이상을 권장합니다. 클라이언트는 Ephemeral Diffie-Hellman(DHE) 또는 Elliptic Curve Ephemeral Diffie-Hellman(ECDHE)과 같은 Perfect Forward Secrecy(PFS)가 포함된 암호 제품군도 지원해야 합니다. Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다.

또한 요청은 액세스 키 ID 및 IAM 위탁자와 관련된 시크릿 액세스 키를 사용하여 서명해야 합니다. 또는 AWS Security Token Service(AWS STS)를 사용하여 임시 자격 증명을 생성하여 요청에 서명할 수 있습니다.

엔드포인트 및 할당량

AWS IoT TwinMaker 엔드포인트 및 할당량

AWS IoT TwinMaker 엔드포인트 및 할당량에 대한 정보는 AWS 일반 참조에서 확인할 수 있습니다.

- 서비스 엔드포인트에 대한 내용은 AWS IoT TwinMaker 서비스 엔드포인트를 참조하십시오.
- 할당량에 대한 내용은 AWS IoT TwinMaker Service Quotas를 참조하십시오.
- API 스로틀링 한도에 대한 내용은 AWS IoT TwinMaker API 스로틀링 한도를 참조하십시오.

AWS IoT TwinMaker 엔드포인트에 대한 추가 정보

에 프로그래밍 방식으로 연결하려면 엔드포인트를 AWS IoT TwinMaker사용합니다. HTTP 클라이언트를 사용하는 경우 다음과 같이 컨트롤 플레인 및 데이터 영역 API에 접두사를 붙여야 합니다. 하지만 필요한 접두사를 자동으로 추가하므로 AWS SDK 및 AWS Command Line Interface 명령에 접두사를 추가할 필요가 없습니다.

- 컨트롤 플레인 API에는 `api` 접두사를 사용하십시오. 예를 들어 `api.iottwinmaker.us-west-1.amazonaws.com`입니다.
- 데이터 영역 API에는 `data` 접두사를 사용하십시오. 예를 들어 `data.iottwinmaker.us-west-1.amazonaws.com`입니다.

AWS IoT TwinMaker 사용 설명서 기록

다음 표에서는 AWS IoT TwinMaker에 대한 문서 릴리스를 소개합니다.

변경 사항	설명	날짜
새 서비스 연결 역할 및 새 IAM 정책	AWS IoT TwinMaker라는 새로운 서비스 연결 역할을 추가했습니다. AWSServiceRoleForIoTTwinMaker AWS IoT TwinMaker다른 서비스를 호출하고 사용자 대신 해당 리소스를 동기화할 수 AWS IoT TwinMaker 있도록 새 AWS 서비스 연결 역할을 추가했습니다. 새 AWSIoTTwinMakerServiceRolePolicy IAM 정책이 이 역할에 연결되며, 이 정책은 사용자를 AWS IoT TwinMaker 대신하여 다른 AWS 서비스를 호출하고 해당 리소스를 동기화할 수 있는 권한을 부여합니다.	2023년 11월 17일
최초 릴리스	AWS IoT TwinMaker 사용 설명서의 최초 릴리스입니다.	2021년 11월 30일

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.