FNT
// simplify complexity

Nutzung, Anreicherung und Konfiguration von Ereignissen für Integrationsszenarien

Filtern und Anreichern von Basisereignissen mit Groovy-Skripten

Senden und Empfangen von angereicherten Ereignissen von und an externe Systeme

Ausimplementieren von kundenspezifischen Regeln für FNT Command

FNT EventEngine

Das leistungsfähige Integrations-Werkzeug für die vollständige Verarbeitung von Events

Als Bestandteil des FNT IntegrationCenter ist die FNT EventEngine darauf spezialisiert, alle Arten von ereignisbasierten Schnittstellen zu verwalten. Insbesondere kann sie verwendet werden, um Ereignisse aus FNT Command zu überwachen, um spezielle Aktionen entweder in FNT Command oder in einer externen Anwendung auszulösen. Diese Ereignisse können gefiltert und mit zusätzlichen Daten aus FNT Command angereichert werden. Die angereicherten Ereignisse können wiederum an andere externe Systeme gesendet oder mit den Daten aus FNT Command über die FNT ReconEngine abgeglichen werden.

FUNKTIONEN

- FNT Command Core-Ereignisse und Ereignisse aus anderen Quellen empfangen
- Filterung eingehender Ereignisse
- Routing eingehender Ereignisse
- Verarbeitung eingehender Ereignisse mit kundenspezifischer Logik
- Anreicherung eingehender Ereignisse mit FNT Command Core-Daten
- Weiterleitung eingehender Ereignisse

EREIGNISGESTEUERTE ANWENDUNGSBEISPIELE

Ereignisgesteuerte Aktualisierungen des FNT Command Kernsystems

Sendet ein externes System ein Ereignis, so können die Daten mithilfe von Groovy-Skripten extrahiert werden. Die erforderlichen Informationen werden an die FNT ReconEngine weitergeleitet, die die Daten entweder automatisch ausrichtet oder einen Delta-Bericht basierend auf vordefinierten Regeln erstellt. Benutzer können dann Änderungen im Delta-Bericht genehmigen und die aktualisierten Daten direkt in der GUI einsehen.

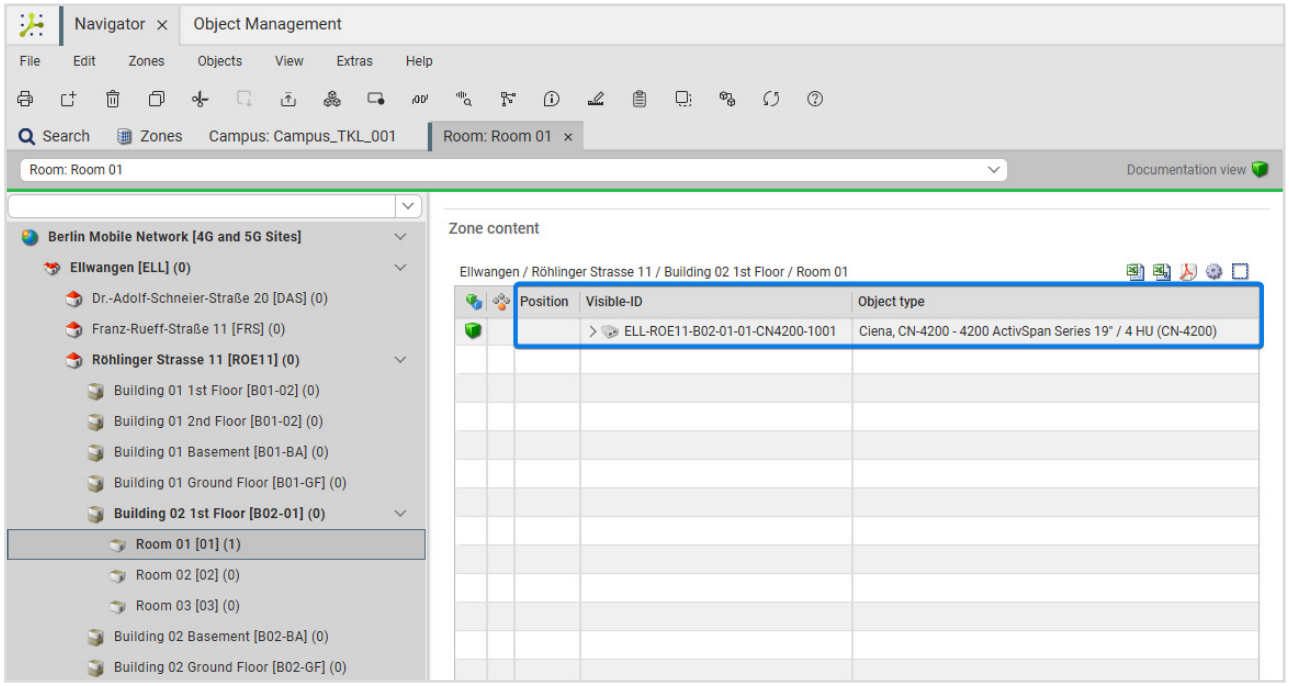

Abbildung 1: Beispielhafte Automatische Generierung der Geräte-VisibleID mit der EventEngine

Datenanreicherung von Ereignissen

Sobald ein externes System ein Ereignis sendet, wie beispielsweise eine Alarmmeldung von einem Monitoring Tool, können die Daten mithilfe von Groovy-Skripten extrahiert werden, wobei zusätzliche Logik wie Filter und Zuordnungen angewendet wird. Ergänzende Informationen, wie beispielsweise Standortadresse, Kabelübersicht, betroffene Kunden und Schutzstatus, können aus FNT Command hinzugefügt werden. Die angereicherten Daten können dann anschließend an ein Ticketsystem weitergeleitet.

```groovy
public class GenerateDeviceVisibleId {

    // Wire beans
    static Logger LOG = LoggerFactory.getLogger(GenerateDeviceVisibleId.class)
    static ZoneService zoneService = ApplicationContextProvider.getBean(ZoneService.class)
    static CampusRepository campusRepo = ApplicationContextProvider.getBean(CampusRepository.class)
    static BuildingRepository buildingRepo = ApplicationContextProvider.getBean(BuildingRepository.class)
    static FloorRepository floorRepo = ApplicationContextProvider.getBean(FloorRepository.class)
    static RoomRepository roomRepo = ApplicationContextProvider.getBean(RoomRepository.class)

    static EventMessageBody apply(Exchange exchange) {
        EventMessageBody event = exchange.getMessage().getMandatoryBody(EventMessageBody.class)
        LOG.info("{}", event)
        IntegrationObjectMessageBody integrationObject = event.getIntegrationObjects().get(0)
        DeviceBase dto = integrationObject.getObject(DeviceBase.class)

        List<Zone> zoneList = zoneService.findZoneByChildElement(dto.getElid())
        LOG.info("Found {} zones for device: {}", zoneList.size(), zoneList)
        if (zoneList.size() != 1) {
            return event
        }
        Zone zone = zoneList.get(0)
```

Abbildung 2: Beispielhaftes Groovy- Skript

Benutzerspezifische Implementierung von Regeln

Erstellt ein Benutzer ein Objekt in FNT Command, beispielsweise einen neuen Service, wird vom System ein grundlegendes Ereignis generiert. Mithilfe von Groovy-Skripten können Filter angewendet werden, wodurch der Prozess bei Bedarf zusätzliche Daten von FNT Command anfordern kann. Benutzerspezifische Logik und Regeln können ebenfalls über Groovy-Skripte *(siehe Abbildung 2)* implementiert werden, wobei die resultierenden Daten wieder in FNT Command gespeichert werden können *(siehe Abbildung 1)*.

Benutzerspezifische, ereignisgesteuerte Aktualisierungen

Platziert ein Benutzer eine Karte in einem Slot, so wird in FNT Command ein grundlegendes Ereignis generiert. Mithilfe von Groovy-Skripten können Filter und Logik angewendet werden, um das Ereignis mit zusätzlichen Daten aus FNT Command anzureichern. Basierend auf definierten Routing-Regeln wird das Ereignis anschließend, beispielsweise, an ein externes System weitergeleitet.

Schnittstellenanpassung (Wrapper)

Wenn ein externer API-Aufruf, wie beispielsweise eine Abfrage von Daten im extern definierten Format, erfolgt, wird dieser Aufruf in einen oder mehrere FNT Command API-Aufrufe umgewandelt. Die angeforderten Daten werden aus FNT Command abgerufen, zurück in das extern definierte Format transformiert und anschließend als Antwort an das externe System gesendet.

DIE WICHTIGSTEN VORTEILE DER FNT EventEngine

- **Definition eigener Logik:** Kunden und Partner können ihre eigene Logik definieren und diese an die entsprechenden Bereiche in der EventEngine weiterleiten.
- **Einfache Interaktion mit in FNT Command gespeicherten Daten:** Die FNT EventEngine verfügt über Adapter zur Integration mit der FNT Command Plattform, die eine einfache Interaktion mit in Command gespeicherten Daten und der ReconEngine ermöglicht. Ereignisse, die von Command ausgelöst werden, können empfangen und verarbeitet werden. Die Command BGE-Komponente ermöglicht das Abfragen und Bearbeiten von Daten sowie den Zugriff auf den ReconEngine NMS-Cache und Synchronisierungsaufträge.
- **Effiziente Verwaltung von Ereignissen:** Die FNT EventEngine dient der Verwaltung aller ereignisbasierten Schnittstellen, um sowohl den Daten- als auch den Ereignisfluss von FNT Command zu externen Systemen und umgekehrt abzuwickeln.
- **Echtzeit-Datenaktualisierungen:** Die EventEngine ermöglicht die Synchronisierung und Aktualisierung von Daten in mehreren Systemen in nahezu Echtzeit, was in Szenarien nützlich ist, in denen eine tägliche vollständige Datensynchronisation nicht ausreicht.
- **Skalierbarkeit für komplexe Anwendungsfälle:** Die EventEngine eignet sich für Szenarien, in denen Daten dynamisch in mehreren Systemen bereitgestellt oder aktualisiert werden müssen, und ist somit für komplexere Anwendungsfälle geeignet, die über traditionelle Batch-Updates hinausgehen.