Live viewer profile generator

Real-time solution accelerator #1

The brief

Build a simple Databricks-style solution accelerator to demonstrate to a Java programmer working for a video streaming site how to build a real-time use case that leverages Snowplow event data.

Requirements

Functional

- Be written in Java (JDK 21)
- Be runnable via Kafka Connect
- Leverage AWS DynamoDB as its state store
- Process sample Snowplow events from our example video streaming site
- Build a live user profile in the state store

Non-functional

- All IP to be fully owned by Snowplow
- Licensed under Apache 2.0 License
- Be made available in a Snowplow-owned GitHub repo
- Contain complete Databricks-style instructions on how to build, deploy, monitor and extend the accelerator

Accelerator design

Scenario

Imagine a video streaming site, similar to Netflix. On this site, viewers can:

1. Start watching a video
2. Potentially pause the video
3. Watch interstitial ads during the video
4. Potentially skip the ad
5. Potentially click on the ad

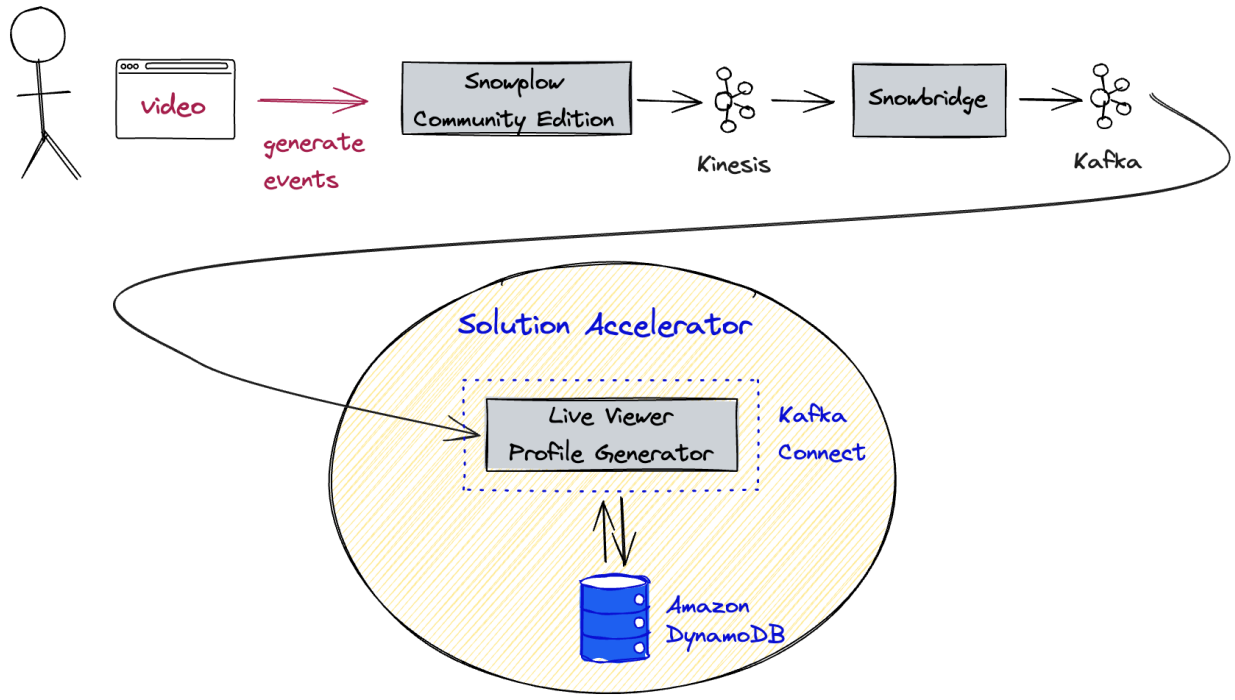
6. Complete or finish the video
7. The cycle repeats

The [example video streaming site] is:

1. Built from React code available in GitHub [here]
2. Using the Snowplow JavaScript Tracker's [media plugin] to emit video and ad events

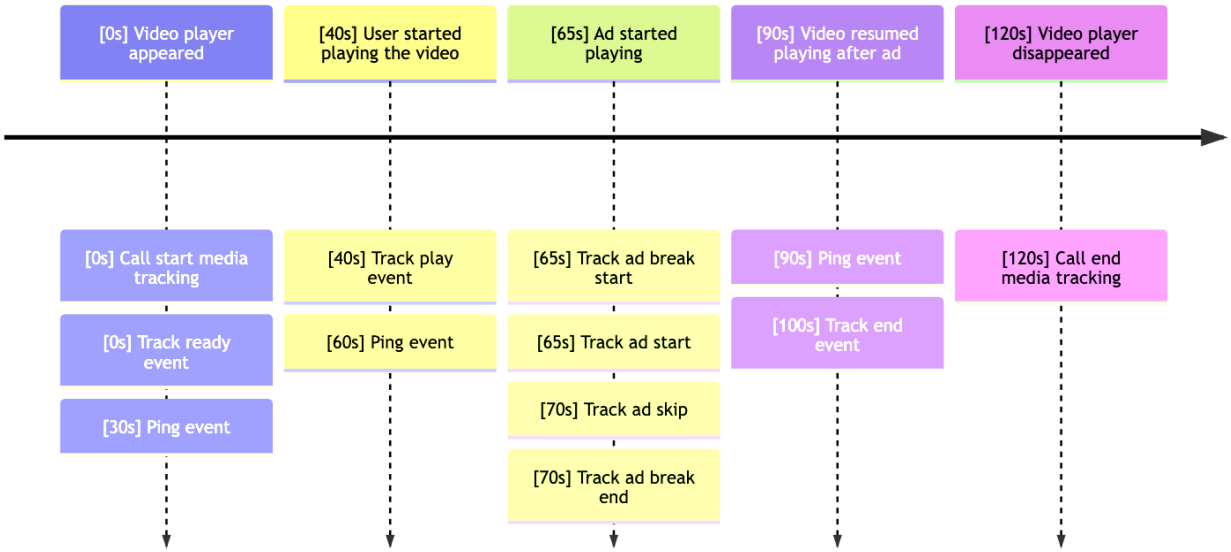
Overall architecture

The following architecture diagram shows the end-to-end data flow, from the example video streaming site through Snowplow and Snowbridge and finally into the accelerator itself:



Snowplow events and entities

A typical media tracking scenario, including emitted events, is detailed in the media plugin documentation, reproduced here:



Ping events are 'heartbeats' emitted to let Snowplow know that the user is still watching the video player.

As you'll see at the bottom of the example video streaming site page, there are four entities (aka objects) that are sent with some or all of the events:

1. Player entity, describing the video player and the specific video hosted within it
2. Session entity, capturing the specific viewing period
3. Ad break entity, capturing a given break of two ads
4. Ad entity, capturing a specific ad

There is no entity for the user watching the videos themselves. In this accelerator we are going to build a live viewer profile from the above events and entities.

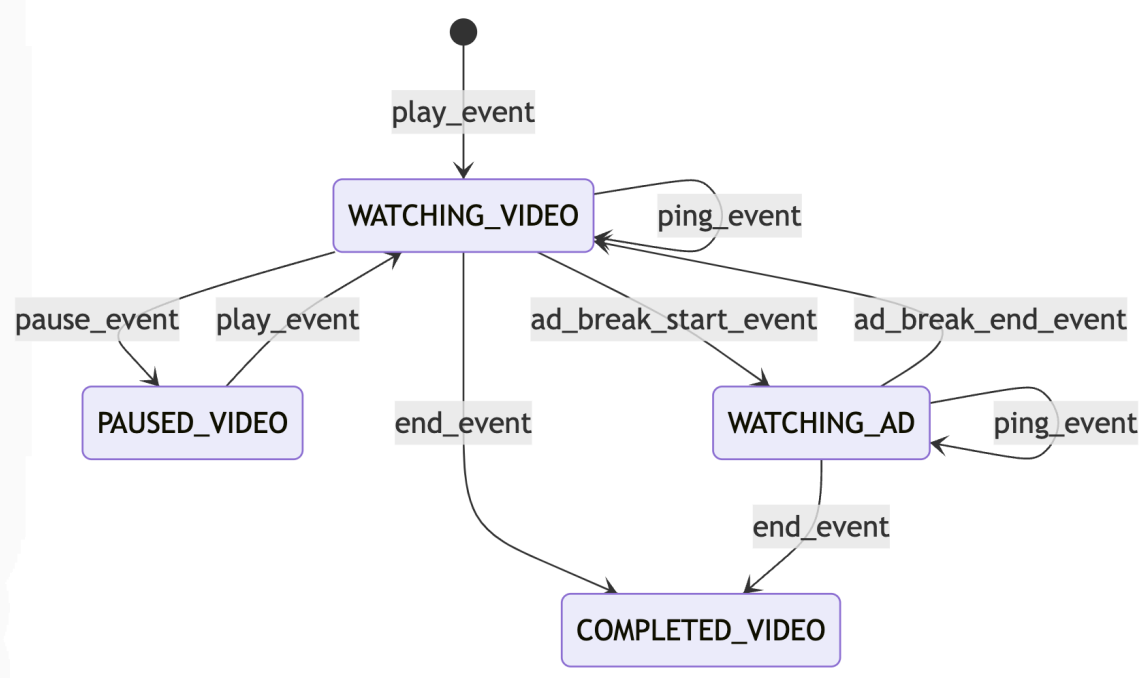
Defining the live viewer profile

Let's define the live viewer profile as a table in DynamoDB that looks like this:

- `viewer_id`, this is taken from the `user_id` field in Snowplow events
- `video_id`, this is the video the viewer is watching
- `status`, this is one of `WATCHING_VIDEO`, `WATCHING_AD`, `PAUSED_VIDEO`, `COMPLETED_VIDEO`
- `ad_id`, this is the advert the viewer is currently watching, only set if status is `WATCHING_AD`
- `video_ts`, the latest timestamp on the video
- `ads_clicked`, a count of how many ads the viewer has clicked on
- `ads_skipped`, a count of how many ads the viewer has skipped

Viewer state transitions

Think of the live viewer status as a simple state machine:



Appendices

Appendix 1: Running Snowplow

- Snowplow Community Edition can be set up on AWS using these instructions
- Snowplow's Snowbridge tool can be used to convert from Amazon Kinesis to Kafka

Appendix 2: Producing media player events

- example video streaming site
- React code available in GitHub here
- The Snowplow JavaScript Tracker's media plugin emits video and ad events

Appendix 3: Consuming the events

Unfortunately there is not a Java consumer library or SDK to make it easier to work with Snowplow events from Java, but here is the Scala consumer library:

- Code on GitHub
- Documentation