

Step 1: Initialization

- In Guide

Initialisation

From v3.11.0 and onwards you need to manually initialize the SDK by calling

```
GameAnalytics.Initialize()
```

from your own GameObject (with script execution order coming after GameAnalytics script's order if your object is in the same scene as the GameAnalytics object as some code is called on Awake event which needs to be called before initializing the sdk).

GameAnalytics supports 5 different types of events: Business, Resource, Progression, Error and Design.

To send an event, remember to include the namespace GameAnalyticsSDK:

```
using GameAnalyticsSDK;
```

The next steps will guide you through the instrumentation of each of the different event types.

- In my Code

A screenshot of a Unity C# script named 'AnalyticalDataStorage'. The script is a MonoBehaviour. It has several 'using' statements at the top: 'using System.Collections;', 'using System.Collections.Generic;', 'using UnityEngine;', 'using GameAnalyticsSDK;', and 'using Facebook.Unity;'. A blue arrow points to the 'using GameAnalyticsSDK;' line. Below the using statements, the class is defined as 'public class AnalyticalDataStorage : MonoBehaviour'. Inside the class, there is a 'private void Awake()' method. A blue arrow points to the 'GameAnalytics.Initialize();' line inside the Awake method. Below it, 'FB.Init();' is also called. The code is highlighted with a light blue background.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using GameAnalyticsSDK;
using Facebook.Unity;

Unity Script (1 asset reference) | 4 references

public class AnalyticalDataStorage : MonoBehaviour
{
    Unity Message | 0 references
    private void Awake()
    {
        GameAnalytics.Initialize();
        FB.Init();
    }
}
```

Step 2: Track Player Progression

- In Guide



Unity SDK - Quick Guide

[← Go back to SDK overview](#)

Track player progression

Use this event to track when players start and finish levels in your game. This event follows a 3 tier hierarchy structure (World, Level and Phase) to indicate a player's path or place in the game.

To add a progression event call the following function:

```
GameAnalytics.NewProgressionEvent(GA_Progression.GAProgressionStatus progressionStatus, string progression01, string progression02, string progression03, int score)
```

[Learn more about the Progression event](#)

- `GameAnalytics.NewProgressionEvent(GA_Progression.GAProgressionStatus progressionStatus, string progression01, string progression02, string progression03, int score)`

- In my code for Day Start Data

```
1 reference
public void dayStartData(int dayDataCount, int customerIncoming)
{
    GameAnalytics.NewProgressionEvent(GAProgressionStatus.Start, "Day : "+(dayDataCount + 1) + " Total Customer : "+ customerIncoming);
    print("DAY START DATA SENT TO _GAME ANALYTICS_");
}
```

At the place of “**string progression01**” I have added “**Day : +(dayDataCount + 1) + ‘ Total Customer’+customerIncoming**” and whole line is giving one single string value


And same for complete status

```
public void dayEndData(int dayDataCount, int customerServed)
{
    GameAnalytics.NewProgressionEvent(GAProgressionStatus.Complete, "Day : "+(dayDataCount + 1) + " Total Customer Served : "+ customerServed);
    print("DAY END DATA SENT TO _GAME ANALYTICS_");
}
```

- I have called this 2 function in my code like this

```
1 reference
IEnumerator StartGame(float t)
{
    dayStartUI.GetComponent<Animator>().Play("In");
    FindObjectOfType<FashionM.Movement.playerMovement>().isWalk = true;
    yield return new WaitForSeconds(t);
    dayStartUI.GetComponent<Animator>().Play("Out");
    FindObjectOfType<AnalyticalDataStorage>().dayStartData(dayCount, (int)customerGoal);
}

```



Before the day starts

```
void Update()
{
    if (manager.CustomerOut >= manager.TotalCustomerGoal && !manager.DayOff)
    {
        manager.isFinalTutorialOver = true;

        try
        {
            FindObjectOfType<SaveGame>().save = 0;
            FindObjectOfType<AnalyticalDataStorage>().dayEndData(manager.dayCount, (int)manager.CustomerOut);
        }
        catch
        {
        }

        StartCoroutine(DayOffLag(0.2f));
        FindObjectOfType<FashionM.Core.AudioManager>().source.PlayOneShot(FindObjectOfType<FashionM.Core.AudioManager>().EndOfDay, WinVolume);
    }
}

```



Just before day complete (End) UI comes