Angry Birds 2: How Audio Changes Everything

Filip Conic
Technical Sound Designer
Rovio

GAME DEVELOPERS CONFERENCE
MARCH 18–22, 2019 | #GDC19


ROVIO


BATTLE
BAY


BATTLE
BAY


ANGRY BIRDS
Match


BATTLE
BAY


ANGRY BIRDS
Match


ANGRY BIRDS
Dream
Blast


ANGRY BIRDS™ 2

Angry Birds Classic

2009

Angry Birds Classic

2009

Angry Birds 2

2015

Angry Birds Classic

2009

Angry Birds 2

2015

Angry Birds The Movie

2016

Angry Birds Classic

2009

Angry Birds 2

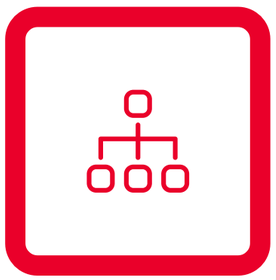
2015

Angry Birds The Movie

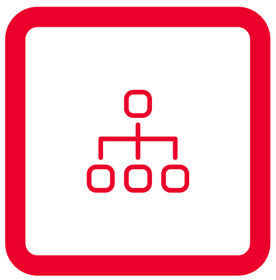
2016

Filip Conic

2017



Scalability



Scalability

Ownership

Scalability

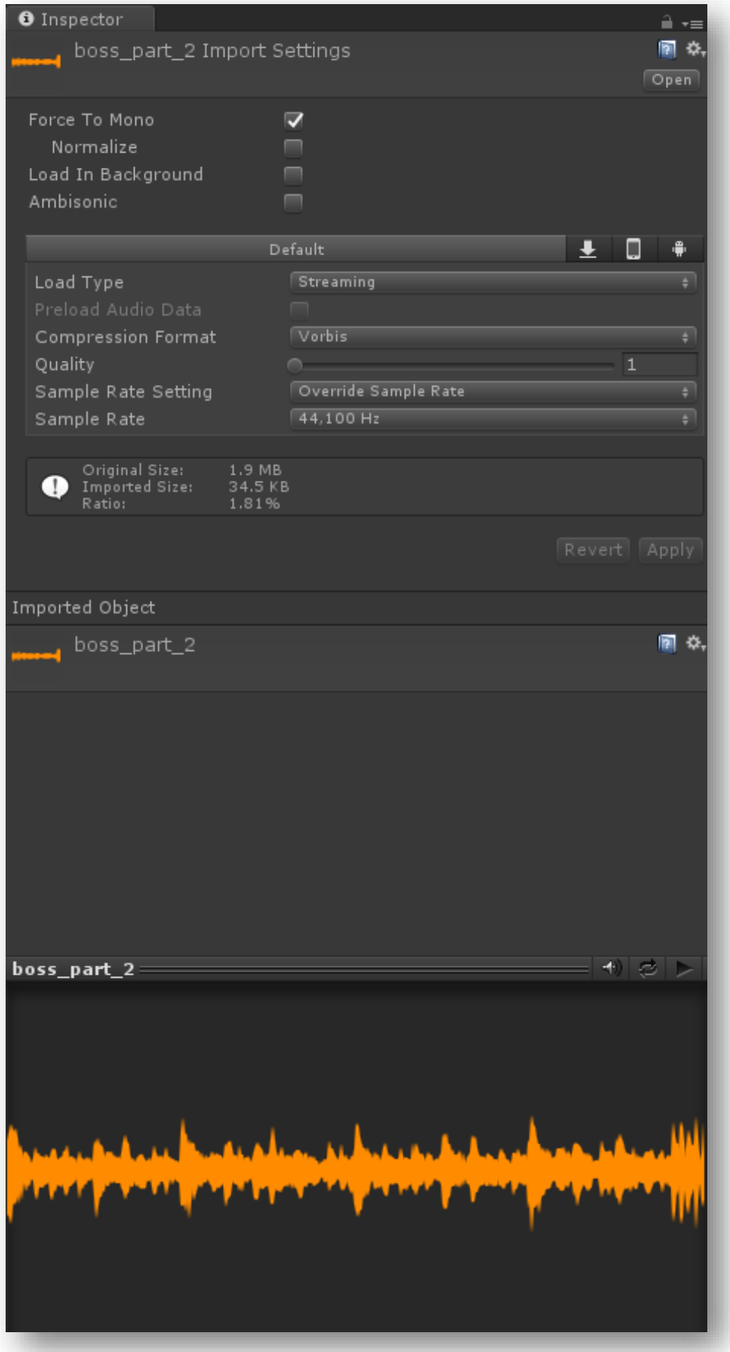
Ownership

Value

10 MB

46 MB

What changed?



```cs
using UnityEngine;
using UnityEditor;


public class AudioSourceImporter : AssetPostprocessor
{
    AudioClip importedAudioClip;
    AudioImporterSampleSettings audioImporterSampleSetting;

    void OnPreprocessAudio()
    {
        importedAudioClip = (AudioClip)AssetDatabase.LoadAssetAtPath(assetImporter.assetPath, typeof(AudioClip));
        var audioImporter = assetImporter as AudioImporter;
        AudioImporterSampleSettings sampleSettings = new AudioImporterSampleSettings();

        if (importedAudioClip != null)
        {
            if (importedAudioClip.name.Contains("music_"))
            {
                audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.Vorbis;
                audioImporterSampleSetting.loadType = AudioClipLoadType.Streaming;
                audioImporterSampleSetting.quality = 1;
            }
            else if (importedAudioClip.name.Contains("ambience_"))
            {
                audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                audioImporterSampleSetting.loadType = AudioClipLoadType.CompressedInMemory;
            }
            else
            {
                if(importedAudioClip.length > 1.5f)
                {
                    audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                    audioImporterSampleSetting.loadType = AudioClipLoadType.DecompressOnLoad;
                }
                else
                {
                    audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                    audioImporterSampleSetting.loadType = AudioClipLoadType.CompressedInMemory;
                }

                // Default settings for assets.
                audioImporter.preloadAudioData = false;

                // Apply SampleSettings to imported asset.
                audioImporter.defaultSampleSettings = sampleSettings;
            }
        }
        else
        {
            // Save assetImporter settings if assetImporter is dirty.
            assetImporter.SaveAndReimport();
        }
    }
}
```

```
using UnityEngine;
using UnityEditor;


public class AudioSourceImporter : AssetPostprocessor
{
    AudioClip importedAudioClip;
    AudioImporterSampleSettings audioImporterSampleSetting;

    void OnPreprocessAudio()
    {
        importedAudioClip = (AudioClip)AssetDatabase.LoadAssetAtPath(assetImporter.assetPath, typeof(AudioClip));
        var audioImporter = assetImporter as AudioImporter;
        AudioImporterSampleSettings sampleSettings = new AudioImporterSampleSettings();

        if (importedAudioClip != null)
        {
            if (importedAudioClip.name.Contains("music_"))
            {
                audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.Vorbis;
                audioImporterSampleSetting.loadType = AudioClipLoadType.Streaming;
                audioImporterSampleSetting.quality = 1;
            }
            else if (importedAudioClip.name.Contains("ambience_"))
            {
                audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                audioImporterSampleSetting.loadType = AudioClipLoadType.CompressedInMemory;
            }
            else
            {
                if(importedAudioClip.length > 1.5f)
                {
                    audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                    audioImporterSampleSetting.loadType = AudioClipLoadType.DecompressOnLoad;
                }
                else
                {
                    audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                    audioImporterSampleSetting.loadType = AudioClipLoadType.CompressedInMemory;
                }

                // Default settings for assets.
                audioImporter.preloadAudioData = false;

                // Apply SampleSettings to imported asset.
                audioImporter.defaultSampleSettings = sampleSettings;
            }
        }
        else
        {
            // Save assetImporter settings if assetImporter is dirty.
            assetImporter.SaveAndReimport();
        }
    }
}
```

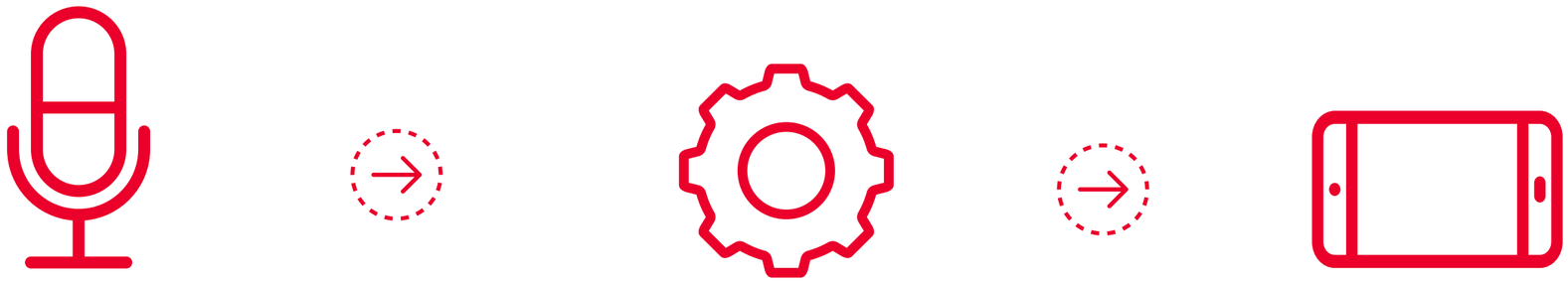
```
using UnityEngine;
using UnityEditor;


public class AudioSourceImporter : AssetPostprocessor
{
    AudioClip importedAudioClip;
    AudioImporterSampleSettings audioImporterSampleSetting;

    void OnPreprocessAudio()
    {
        importedAudioClip = (AudioClip)AssetDatabase.LoadAssetAtPath(assetImporter.assetPath, typeof(AudioClip));
        var audioImporter = assetImporter as AudioImporter;
        AudioImporterSampleSettings sampleSettings = new AudioImporterSampleSettings();

        if (importedAudioClip != null)
        {
            if (importedAudioClip.name.Contains("music_"))
            {
                audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.Vorbis;
                audioImporterSampleSetting.loadType = AudioClipLoadType.Streaming;
                audioImporterSampleSetting.quality = 1;
            }
            else if (importedAudioClip.name.Contains("ambience_"))
            {
                audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                audioImporterSampleSetting.loadType = AudioClipLoadType.CompressedInMemory;
            }
            else
            {
                if(importedAudioClip.length > 1.5f)
                {
                    audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                    audioImporterSampleSetting.loadType = AudioClipLoadType.DecompressOnLoad;
                }
                else
                {
                    audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                    audioImporterSampleSetting.loadType = AudioClipLoadType.CompressedInMemory;
                }

                // Default settings for assets.
                audioImporter.preloadAudioData = false;

                // Apply SampleSettings to imported asset.
                audioImporter.defaultSampleSettings = sampleSettings;
            }
        }
        else
        {
            // Save assetImporter settings if assetImporter is dirty.
            assetImporter.SaveAndReimport();
        }
    }
}
```

```
using UnityEngine;
using UnityEditor;


public class AudioSourceImporter : AssetPostprocessor
{
    AudioClip importedAudioClip;
    AudioImporterSampleSettings audioImporterSampleSetting;

    void OnPreprocessAudio()
    {
        importedAudioClip = (AudioClip)AssetDatabase.LoadAssetAtPath(assetImporter.assetPath, typeof(AudioClip));
        var audioImporter = assetImporter as AudioImporter;
        AudioImporterSampleSettings sampleSettings = new AudioImporterSampleSettings();

        if (importedAudioClip != null)
        {
            if (importedAudioClip.name.Contains("music_"))
            {
                audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.Vorbis;
                audioImporterSampleSetting.loadType = AudioClipLoadType.Streaming;
                audioImporterSampleSetting.quality = 1;
            }
            else if (importedAudioClip.name.Contains("ambience_"))
            {
                audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                audioImporterSampleSetting.loadType = AudioClipLoadType.CompressedInMemory;
            }
            else
            {
                if(importedAudioClip.length > 1.5f)
                {
                    audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                    audioImporterSampleSetting.loadType = AudioClipLoadType.DecompressOnLoad;
                }
                else
                {
                    audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                    audioImporterSampleSetting.loadType = AudioClipLoadType.CompressedInMemory;
                }

                // Default settings for assets.
                audioImporter.preloadAudioData = false;

                // Apply SampleSettings to imported asset.
                audioImporter.defaultSampleSettings = sampleSettings;
            }
        }
        else
        {
            // Save assetImporter settings if assetImporter is dirty.
            assetImporter.SaveAndReimport();
        }
    }
}
```

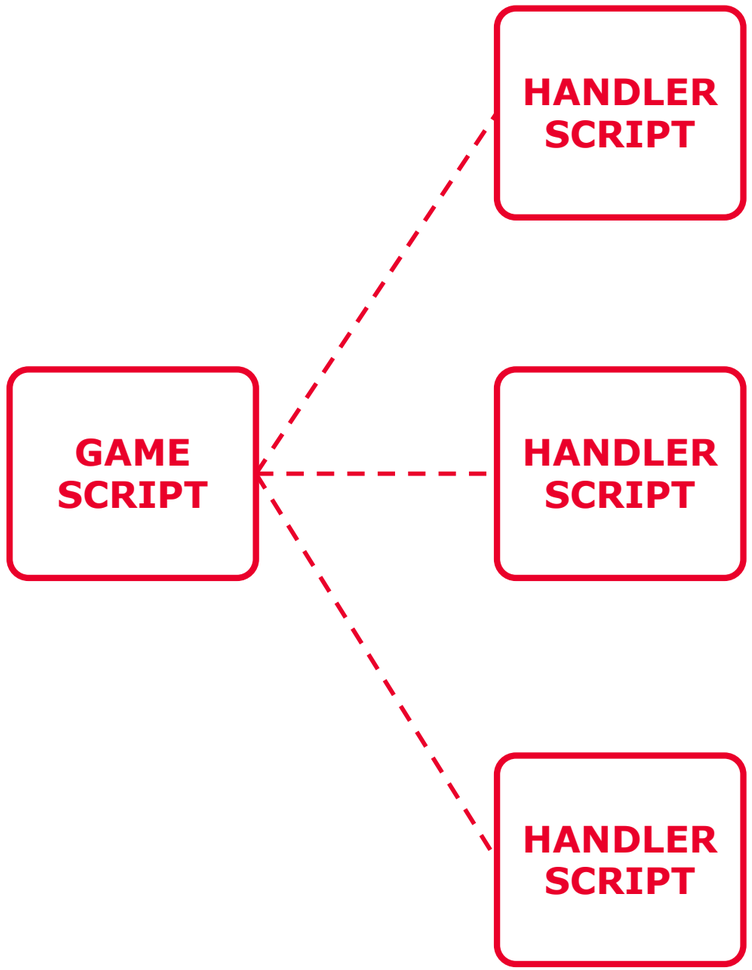
```
using UnityEngine;
using UnityEditor;


public class AudioSourceImporter : AssetPostprocessor
{
    AudioClip importedAudioClip;
    AudioImporterSampleSettings audioImporterSampleSetting;

    void OnPreprocessAudio()
    {
        importedAudioClip = (AudioClip)AssetDatabase.LoadAssetAtPath(assetImporter.assetPath, typeof(AudioClip));
        var audioImporter = assetImporter as AudioImporter;
        AudioImporterSampleSettings sampleSettings = new AudioImporterSampleSettings();

        if (importedAudioClip != null)
        {
            if (importedAudioClip.name.Contains("music_"))
            {
                audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.Vorbis;
                audioImporterSampleSetting.loadType = AudioClipLoadType.Streaming;
                audioImporterSampleSetting.quality = 1;
            }
            else if (importedAudioClip.name.Contains("ambience_"))
            {
                audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                audioImporterSampleSetting.loadType = AudioClipLoadType.CompressedInMemory;
            }
            else
            {
                if(importedAudioClip.length > 1.5f)
                {
                    audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                    audioImporterSampleSetting.loadType = AudioClipLoadType.DecompressOnLoad;
                }
                else
                {
                    audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                    audioImporterSampleSetting.loadType = AudioClipLoadType.CompressedInMemory;
                }

                // Default settings for assets.
                audioImporter.preloadAudioData = false;

                // Apply SampleSettings to imported asset.
                audioImporter.defaultSampleSettings = sampleSettings;
            }
        }
        else
        {
            // Save assetImporter settings if assetImporter is dirty.
            assetImporter.SaveAndReimport();
        }
    }
}
```

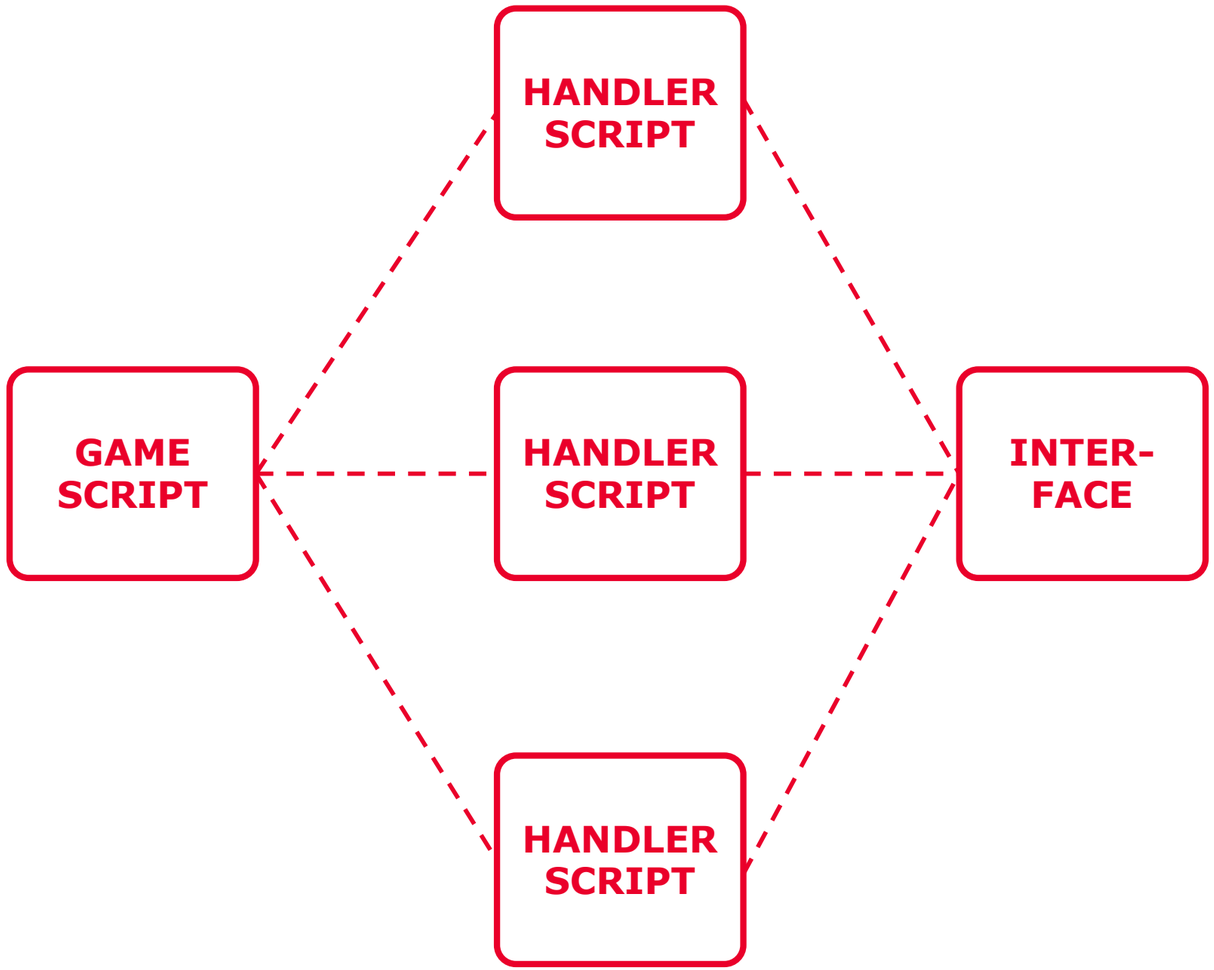
```
using UnityEngine;
using UnityEditor;


public class AudioSourceImporter : AssetPostprocessor
{
    AudioClip importedAudioClip;
    AudioImporterSampleSettings audioImporterSampleSetting;

    void OnPreprocessAudio()
    {
        importedAudioClip = (AudioClip)AssetDatabase.LoadAssetAtPath(assetImporter.assetPath, typeof(AudioClip));
        var audioImporter = assetImporter as AudioImporter;
        AudioImporterSampleSettings sampleSettings = new AudioImporterSampleSettings();

        if (importedAudioClip != null)
        {
            if (importedAudioClip.name.Contains("music_"))
            {
                audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.Vorbis;
                audioImporterSampleSetting.loadType = AudioClipLoadType.Streaming;
                audioImporterSampleSetting.quality = 1;
            }
            else if (importedAudioClip.name.Contains("ambience_"))
            {
                audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                audioImporterSampleSetting.loadType = AudioClipLoadType.CompressedInMemory;
            }
            else
            {
                if(importedAudioClip.length > 1.5f)
                {
                    audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                    audioImporterSampleSetting.loadType = AudioClipLoadType.DecompressOnLoad;
                }
                else
                {
                    audioImporterSampleSetting.compressionFormat = AudioCompressionFormat.ADPCM;
                    audioImporterSampleSetting.loadType = AudioClipLoadType.CompressedInMemory;
                }

                // Default settings for assets.
                audioImporter.preloadAudioData = false;

                // Apply SampleSettings to imported asset.
                audioImporter.defaultSampleSettings = sampleSettings;
            }
        }
        else
        {
            // Save assetImporter settings if assetImporter is dirty.
            assetImporter.SaveAndReimport();
        }
    }
}
```

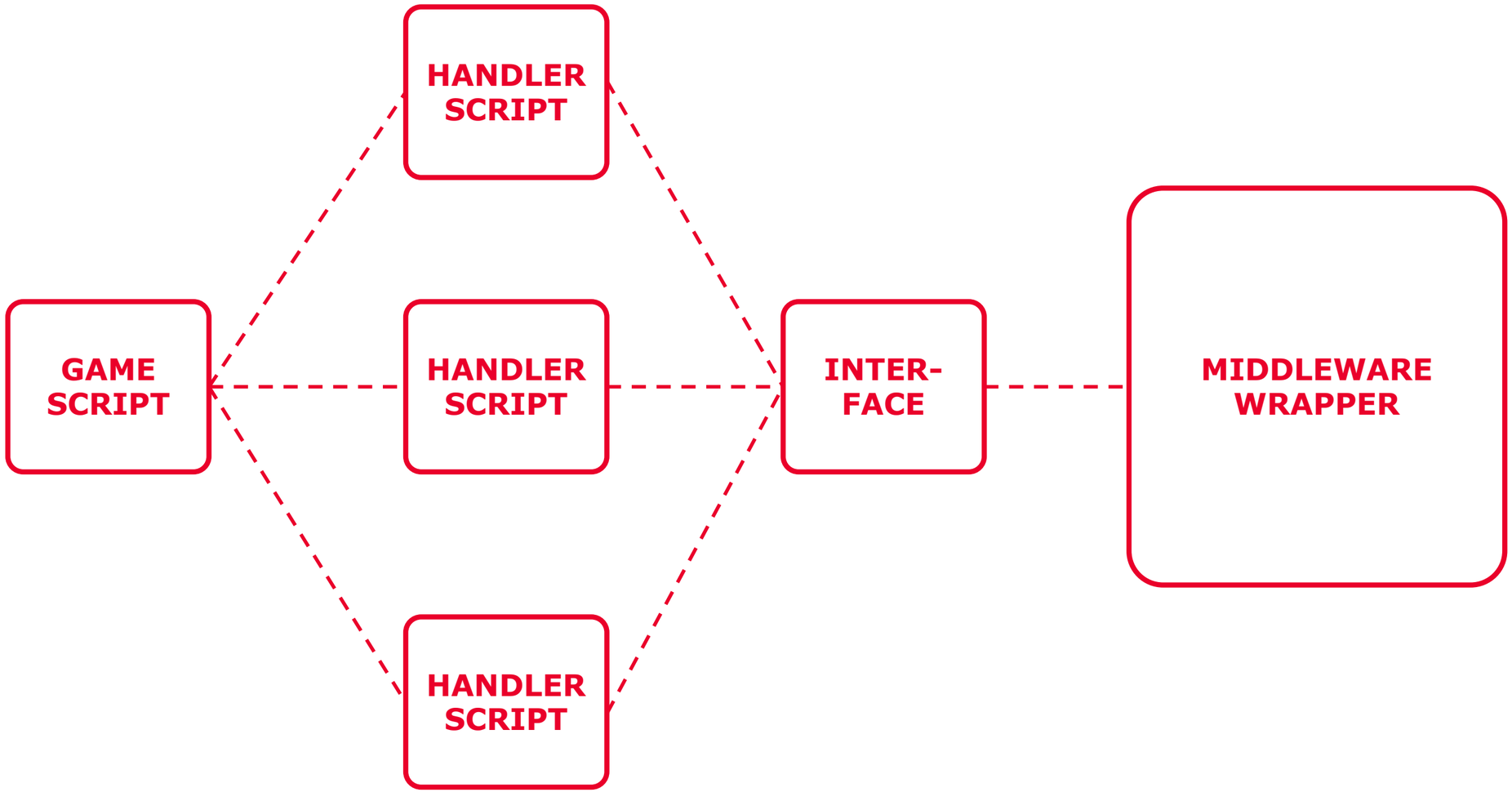
Owning the Audio Pipeline

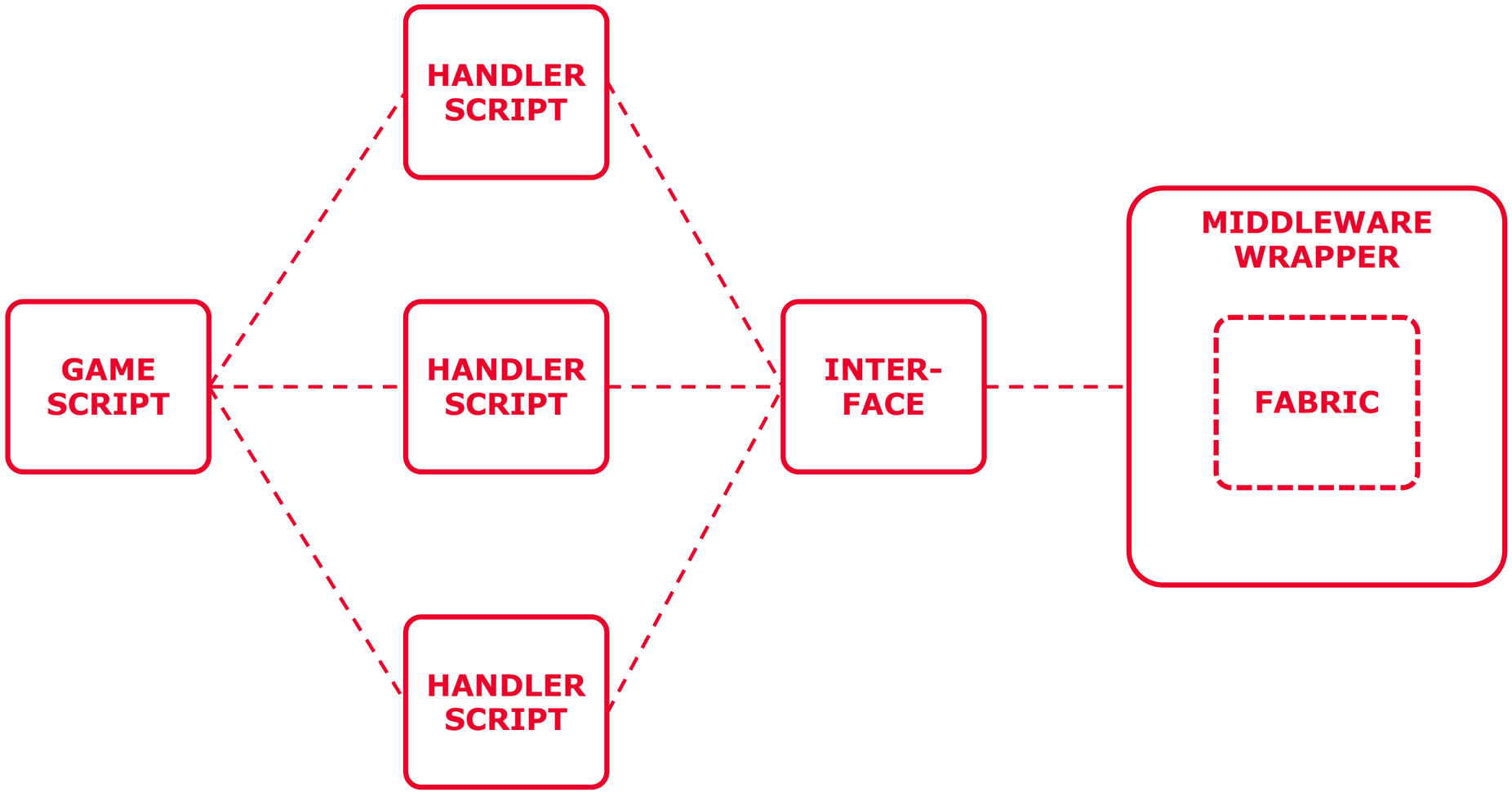


GAME
SCRIPT


HANDLER SCRIPT
GAME SCRIPT
HANDLER SCRIPT
HANDLER SCRIPT

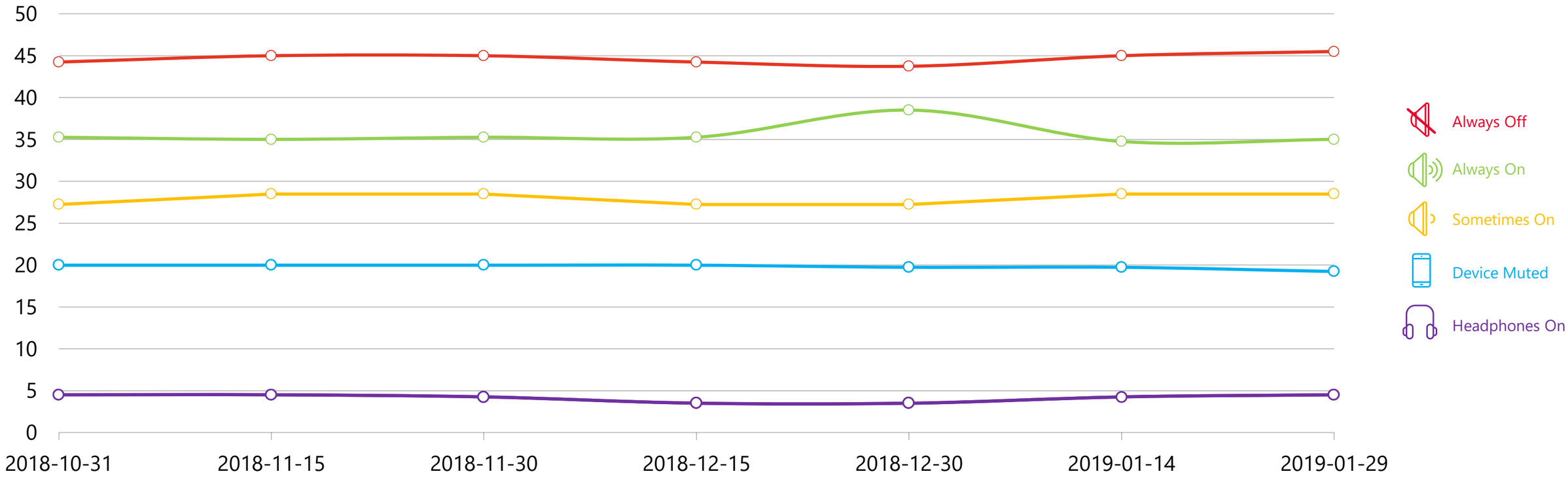

HANDLER SCRIPT
GAME SCRIPT
HANDLER SCRIPT
INTER-FACE
HANDLER SCRIPT


HANDLER SCRIPT
GAME SCRIPT
HANDLER SCRIPT
INTER-FACE
MIDDLEWARE WRAPPER
HANDLER SCRIPT

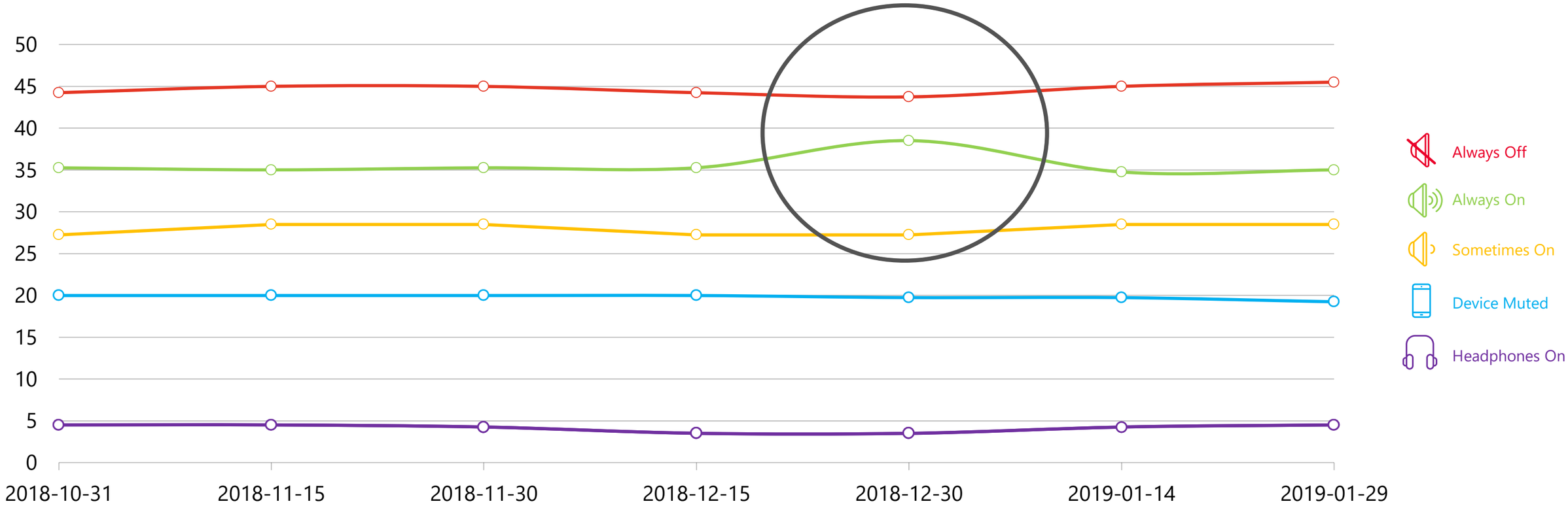

HANDLER SCRIPT
GAME SCRIPT
HANDLER SCRIPT
INTER-FACE
MIDDLEWARE WRAPPER
FABRIC
HANDLER SCRIPT


SCORE: 433
0

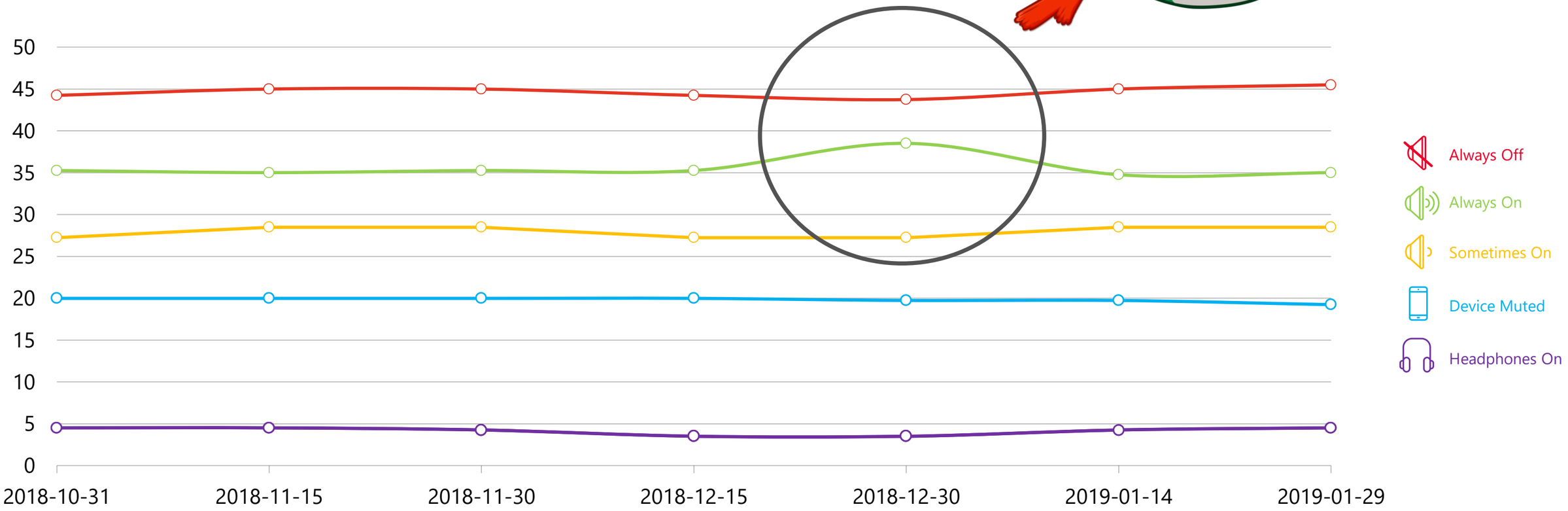
TIMESERIES OF AUDIO USAGE (SHARE OF DAU)

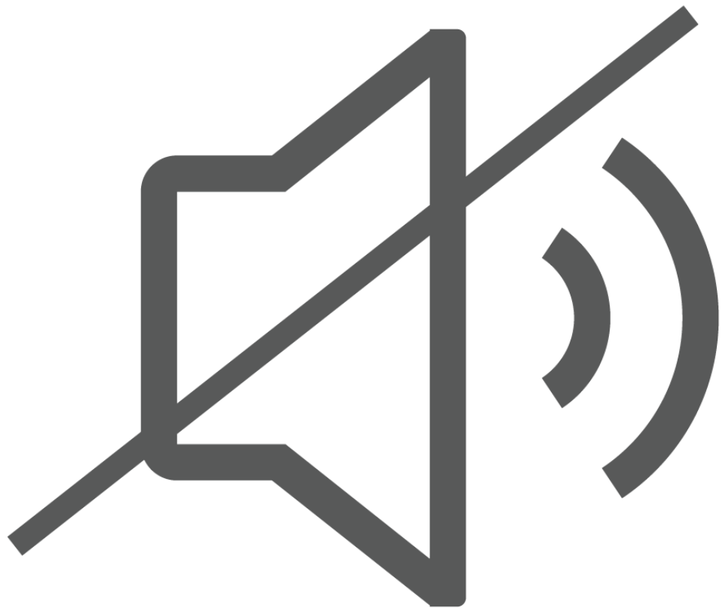


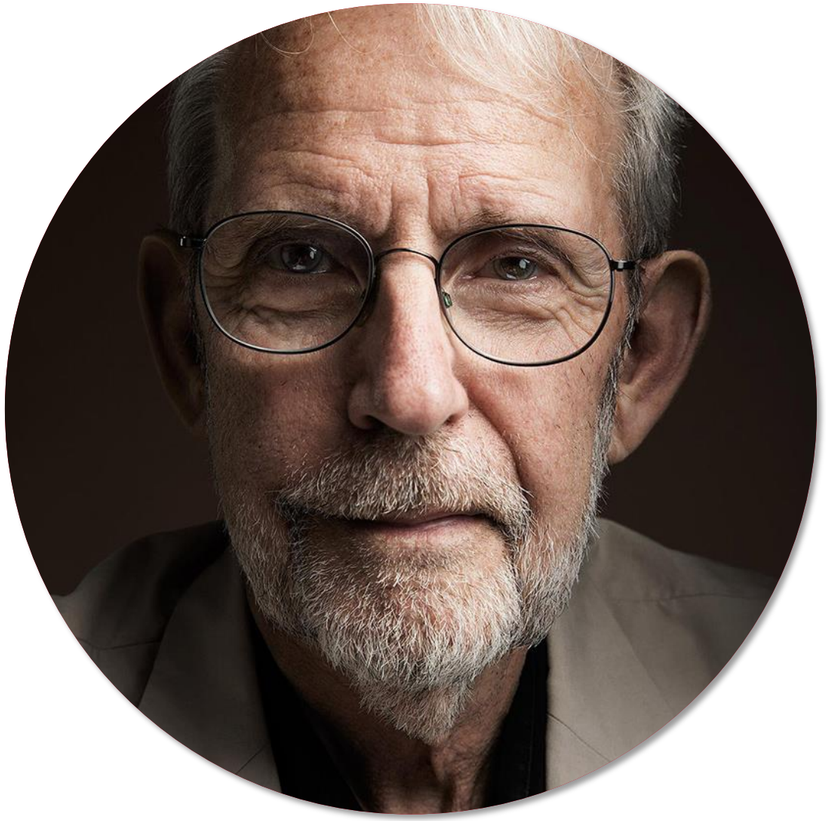
TIMESERIES OF AUDIO USAGE (SHARE OF DAU)



TIMESERIES OF AUDIO USAGE (SHARE OF DAU)







Walter Murch

Dense Clarity – Clear Density

Language
Encoded

Violet Yellow Red

Language
Encoded

Sound Effects
Encoded - Embodied

Music
Embodied

Violet

Yellow

Red

Warm toned sounds

Cool toned sounds

Key engagement sounds

Scalability

Ownership

Value

@filipconic @filipconic

filip.conic@rovio.com

ROVIO