CN Arcade Process / Best Practices

for

Cartoon Network Digital / MSS

Version 1.0

Prepared by: WarnerMedia MSS

Table of Contents

1 INTRODUCTION 4
1.1 DOCUMENT PURPOSE 4
1.2 INTENDED AUDIENCE 4
1.3 DEFINITIONS, ACRONYMS, AND ABBREVIATIONS 4
1.4 WHAT IS THE CN ARCADE APP ? 7
2 TECHNICAL COMPONENTS 8
2.1 API 8
2.1.1 API Service Dependencies 8
2.1.2 API Git Repo 9
2.1.3 API Local environment set up 9
2.1.4 Test, Acceptance and Production Environment(s) Set-Up 9
2.1.5 CN Arcade API Domain URLs: 10
2.1.6 API Development workflow 10
2.2 JENKINS INSTANCES 11
2.3 CMS 11
2.3.1 Setting up local CMS environments 11
2.3.2 CMS Acceptance 11
2.3.3 CMS Production 12
2.3.4 CMS Producer information 12
2.4 UNITY 12
2.4.1 Using the Unity Editor 12
2.4.2 Unity Build Process (UCB) 13
2.4.3 Unity Architecture 14
2.4.4 Unity SDK's 14
2.4.5 Unity Version Control 15
3 ACR INGESTION PROCESS 16
3.1 GIT REPO LOCATION 17
3.2 AWS CLOUD RESOURCES 17
3.3 TOOLS NEEDED TO PROCESS ACR CONTENT 18
3.4 ACR AUTOMATIC CONTENT INGESTION PROCESS DIAGRAM 18
3.5 APPLICATION WORKFLOW 19
3.6 ACR CONTACT INFORMATION 19
4 CN ARCADE STUNTS 20
4.1 SDK DEVELOPMENT 15
4.2 DEVELOPMENT ENVIRONMENTS 21
4.3 CREATE STUNT ENTITY IN CMS 22
4.4 CREATE A TEST USER 23
4.5 ADDING SCRIPTS TO STUNT PAGE 23
4.6 CONFIGURING SDK FOR LOCAL DEVELOPMENT 23
4.7 TESTING INITIALIZATION 24
4.8 TROUBLESHOOTING INITIALIZATION 25
4.9 USING SDK 26
4.10 EXAMPLE: UNLOCKING ACHIEVEMENT 26
4.11 PRE-DEPLOYMENT REVIEW AND QA 27
4.12 DEPLOYING A STUNT 28
4.13 SOURCE CODE 29
5 WORK REQUEST AND APPROVAL PROCESS 29
5.1 REQUESTING WORK 30
5.1.1 Creating a JIRA request 30

5.1.2 Discovery meeting 30
5.1.3 Level of Effort (LOE) 31
5.1.4 LOE review and CN producer approval 31
5.1.5 MSS QA Verification 31
5.1.6 Stakeholder approval and resource assignments 32
5.1.7 CN approval for production deployment 32
5.1.8 Change Management process for production changes 33

6 SYSTEM INTEGRATIONS 33

7 MONITORING 34

7.1 EMERGENCY RESPONSE PROCEDURE 34
7.2 BIG PANDA 34
7.3 PRODUCTION ESCALATION PROCESS 35

8 URL LINKS 35

THIS SECTION WILL SERVE AS A QUICK REFERENCE GUIDE FOR URL LINKS FOUND THROUGHOUT THE DOCUMENT. 35
8.1 API GIT REPO 35
8.2 CN ARCADE API DOMAIN URLS: 35
8.3 JENKINS INSTANCES 35
8.4 SETTING UP LOCAL CMS ENVIRONMENTS 36
8.5 CMS ACCEPTANCE 36
8.6 CMS PRODUCTION 36
8.7 CMS PRODUCER INFORMATION 36
8.8 CREATE A TEST USER 36
8.9 CREATING A JIRA REQUEST TICKET 36
8.10 LEVEL OF EFFORT (LOE) TEMPLATE 37
8.11 LOE'S IN REVIEW AND RESOURCE ASSIGNMENTS 37
8.12 CHANGE MANAGEMENT REQUEST FORM 37
8.13 CN EMERGENCY RESPONSE PROCEDURE 37
8.14 CN BIG PANDA ALERTS LIST 37
8.15 SYSTEMS THAT INTEGRATE WITH BIG PANDA 37

1 Introduction

1.1 Document Purpose

This document outlines the MSS processes associated with development for the CN Arcade application. It should be noted that this is a living internal document. This document is to be used by the development team to help understand agreed-upon processes defined by MSS, Media Monks, and CN Stakeholders.

1.2 Intended Audience

This document is intended for all developers and stakeholders of the mobile games/software *CN Arcade* at Cartoon Network.

1.3 Definitions, Acronyms, and Abbreviations

ACR – (Automatic Content Recognition) An identification technology to recognize content played on a media device or present in a media file. ... For example, developers of the application can then provide personalized complementary content to viewers.

ACRCloud (Formerly Syntec TV) is an automatic content recognition platform based on acoustic fingerprinting technology. Its creator intended to help media, broadcasters and app developers to identify, monitor and monetize content on the second screen.

Adobe Analytics – A basic analytics package for the CN Arcade. Analytics events are collected on the Adobe Analytics dashboard.

API - (Application Programming Interface) A set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, or other service.

Application - This refers to the CN Arcade application.

Argo - Deployment dashboard used to deploy builds created by Bamboo.

Bamboo - Continuous integration deployment tool that facilitates builds for the backend services.

Big Panda- WarnerMedia's Digital On Call system used to monitor application alerts.

Charles Proxy – A web proxy (HTTP Proxy / HTTP Monitor) that runs on your own computer. Your web browser (or any other Internet application) is then configured to access the Internet through Charles, and Charles is then able to record and display for you all of the data that is sent and received.

CMS – (Content Management System) CMS is software / application that uses a database to manage all content.

CN - (Cartoon Network). The brand name for Cartoon Network Digital.

CN Arcade – The CN Arcade is a mobile application for iOS and Android platforms.

Content Ingestion – CN Arcade import and create an ACR thumbprint for all new CN Arcade content.

Conviva - The application's analytics API for video measure and analytics architecture.

DFP - (DoubleClick For Publishers). This is the Google ads SDK that facilitates display ad server functionality.

DOC - (Digital-On-Call Center) 24/7 monitoring and escalation team that oversees production systems.

ERP (Emergency Response Procedure) - Instructions for how to respond to production alerts/issues reported to the DOC.

FreeWheel - An incentivized video ads SDK video ad server.

Gradle – an open-source build-automation system that builds upon the concepts of Apache Ant and Apache Maven and introduces a Groovy-based domain-specific language (DSL) instead of the XML form used by Apache Maven for declaring the project configuration.

Jenkins – a free and open source automation server. Jenkins helps to automate the non-human part of the software development process, with continuous integration and facilitating technical aspects of continuous delivery.

Kochava – An advertisement campaign attribution SDK for Unity. Accredits app installs from advertisements to Kochava and Unity Analytics dashboard.

LINQ – (Language Integrated Query) A set of technologies based on the integration of query capabilities directly into the C# language.

LOE - (Level of Effort) This refers to the document that MSS developers must deliver when giving their estimate of tasks and time needed to complete requested work made by CN.

MediaMonks - 3rd party application development company that partners with CN/MSS to develop the application.

MediaMonks – This is the vendor that CN uses for a majority of software development for CN Arcade.

MSS – (Media Software Services). This is the name of the professional software services development team within WarnerMedia.

Node JS - An open source development platform for executing server-side JavaScript code.

PlayerPrefs – (aka Player Preferences). Unity's local saving system where a user's preferences are stored between sessions.

PostGres - (aka PostGresSQL) a free and open-source relational database management system (RDBMS) emphasizing extensibility and technical standards compliance.

PP/ToS – (Privacy Policy and Terms of Service)

PR – (Pull Request). A proposed change to code in a repository. Requires approval of another developer to merge the change.

ReadMe - Developer documentation found in the repo that includes information about the software, such as installation and configuration instructions.

Reative Extensions – (a.k.a. ReactiveX) A set of tools allowing imperative programming languages to operate on sequences of data regardless of whether the data is synchronous or asynchronous.

Repo - (a.k.a. Repository; Git Repo) The online, version-controlled storage of any software that is developed by/for CN.

S3 – (Simple Storage Service) This is an Amazon service that is a scalable, high-speed, web-based cloud storage service designed for online backup and archiving of data and applications on Amazon Web Services.

SDK - (Software Development Kit). The games developed at CN use many SDKs to facilitate the various features that CN needs to implement. Some of these help with ads, others help with privacy policies, while others help with core functionalities of the game. For the purposes of this document we will list the ones that are common to all games and software at CN.

SettingsInstaller – A Unity asset that facilitates game settings manipulation, including state changes, environment switching, and different parts of the code.

UCB – (Unity Cloud Build). This provides continuous integration services for Unity projects. It automates the process of creating builds on Unity's servers which helps developers catch problems sooner, share builds with collaborators, and iterate versions of development more rapidly.

UniRx – A plugin for Unity (see Reative Extensions) that allows LINQ to events and more. This is free and opensource on GitHub.

Unity Analytics - Unity's built in Analytics service tool. Sends custom events to Unity Analytics dashboard.

UniWebView – A Unity plugin that facilitates the presentation of web content within a Unity game/app.

Vendor – Any outside studio, business, developer(s), or other individuals that deliver products of services on a contractual basis with/for CN.

Webpack - an open-source JavaScript module bundler. It is a module bundler primarily for JavaScript, but it can transform front-end assets like HTML, CSS, and images if the corresponding loaders are included.

ZenDesk - The customer service management tool utilized by the application.

Zenject - A lightweight highly performant dependency injection framework built specifically to target Unity 3D (however it can be used outside of Unity as well). It can be used to turn your application into a collection of loosely coupled parts with highly segmented responsibilities

1.4 What is the CN Arcade App ?

Cartoon Network Arcade is a mobile application enabling fans to participate with Cartoon Network more directly through an arcade of games, collection of figures, gamified voting, and trivia contests. With unique avenues designed to increase fandom and lock-in user retention via fresh content and a sense of ownership, participation, and personalization. Users will have access to play a vast collection of games with a variety of genres and IP. Registration won’t be a requirement to play these games, but fans will be encouraged to create an account to secure their progress.

2 Technical Components

The CN Arcade is a mobile application for iOS and Android platforms built using the Unity3D game development engine. The games and stunts served from within the app are hosted remotely and presented via mobile WebView. Digital content for the app is uploaded by producers through a custom CMS. CMS content is exposed to the app through a custom API.

2.1 API

The CN Arcade API is a backend API service that supports the application.

The API supports CMS editorial content, game content, user achievement data, user game progress, and user scores.

The API is developed in Typescript which is compiled to Node.JS. It is designed to use AWS serverless architecture, utilizing the AWS lambda service. MediaMonks and MSS share the development work.

2.1.1 API Service Dependencies

This API service uses AWS in-memory storage Amazon ElastiCache for Redis service to access and manage application data and user data.

AWS Kinesis Stream service is used to stream handle commands, as well as data and event streams.

2.1.2 API Git Repo

The version-controlled source code for CNArcade is stored and tracked in the Git Repo located in the following location:

- https://github.com/turnercode/cartoon-play-api

2.1.3 API Local environment set up

To develop and run the API project locally, the developer needs to install and run the following:

- Local kinesis service:
- npm i kinesalite --save-dev
- ./node_modules/.bin/kinesalite --path tmp/
- Local Redis service
- brew install redis
- redis-server --protected-mode no
- Generate serverless config for environment
- ./node_modules/.bin/gulp --env=dev serverless-config
- Run serverless offline
- REDIS_PORT=6379 REDIS_DB=0 make runslsdev

2.1.4 Test, Acceptance and Production Environment(s) Set-Up

The API application is deployed to the AWS environments with Jenkins continuous integration and deployment tool. The Jenkins access credential is stored in an AWS secret manager. See onboarding documentation for AWS secret manager information.

2.1.5 CN Arcade API Domain URLs:

Test: https://api-test.cartoonnetworkarcade.com/api/{apiPath+}

Acceptance: https://api-acc.cartoonnetworkarcade.com/api/{apiPath+}

Production: https://api.cartoonnetworkarcade.com/api/{apiPath+}

2.1.6 API Development workflow

- All new features require its own branch for development. This new branch should always be branched from the develop branch.
- All branch name follows GitFlow conventions:
    - Feature/cp-xxxx.
- After a successful local test of the new. feature, the developer must submit a new PR for merging the feature branch to the develop branch.
- At least two (2)developers must approve. The PR.
- Once the PR is approved it is merged to the develop branch.
    - Coordinate with developers and QA for deployment to test environment.
    - This is done by emailing all team/stakeholders. See section 6.1.6 for approval process details.
- Once a new release is scheduled, a release branch is created from the develop branch. This will include all of the features to be included, and will then be deployed to the acceptance environment for QA testing.
- Approval is needed by QA and The Brand before the release branch is merged to the master branch and deployed to the production environment.

2.2 Jenkins Instances

Jenkins is used to build and test software projects, making it easier for developers to integrate changes.

- Production
    - https://cartoon-network-cn-play-app-prod.sysops.us-east-1.321061860761.monkapps.com/jenkins
      Jenkins username: build
      Jenkins password:
- Acceptance
    - http://cartoon-network-cn-play-app-acc.sysops.us-east-1.321061860761.monkapps.com/jenkins
      Jenkins username: build
      Jenkins password:
- Testing
    - http://cartoon-network-cn-play-app-test.sysops.us-east-1.321061860761.monkapps.com/jenkins
      Jenkins username: build
      Jenkins password:

2.3 CMS

CN Arcade CMS provides a backend framework that delivers content to CN Arcade.

2.3.1 Setting up local CMS environments

Developer(s) will need to setup and run the CN Arcade CMS on their local machine. Information on environment setup is located in the CMS GitHub repo (readme file). GitHub Repo location can be found here:

- https://github.com/turnercode/cartoon-play-cms

2.3.2 CMS Acceptance

After the developer(s) have made changes locally, they must deploy their changes to the CN Video CMS acceptance environment.

- https://cms-acc-l3d9h0i366.cartoonnetworkarcade.com/

2.3.3 CMS Production

Once changes from the acceptance environment have been approved and merged, the new code will be ready for the production environment. *All production deployments must follow the approval process outlined in section 5.1.6 of this document.*

- https://cms-p5z8zbvsz0.cartoonnetworkarcade.com/dn7bUiCG8v3y5ry9/login

2.3.4 CMS Producer information

CN Producers have created a confluence page which provides details on how to set up stunts, feeds, loot tables, and games. In addition, this page provides helpful instructions for completing CMS administrator tasks.

- https://docsprod.turner.com/pages/viewpage.action?spaceKey=CPPA&title=CMS+Information

2.4 Unity

The README file for CN Arcade has technical tips and best practices.

Link to README file is in the root of the CN Arcade Unity repository.

2.4.1 Using the Unity Editor

The CN Arcade Unity editor contains features that facilitate use that aren’t obvious in the contained README file. These features are as follows:

- Switching environments can be done by using the Tools > MM dropdown in the top navigation bar and picking an environment. By doing so, this will wipe the PlayerPrefs and assigns a new guest token.
- A Debug GUI with buttons and input fields exists to make it easier to test features and simulateAPI data. It can only run while in the Unity editor is in Play mode.
- The SettingsInstaller asset needs to be modified when making a new AppState or AppSection. Sometimes pulling in new features can pull incorrect SettingsInstaller medtadata. If this happens, quitting and restarting Unity, as well as discarding local meta files, can fix SettingsInstaller bugs.
- UniWebview may not show up when clicking a game in the feed. Change the platform logic in the WebViewContainer script to include UNITY_EDITOR, so that you can open games in the WebView on a Mac.

2.4.2 Unity Build Process (UCB)

Unity Cloud Build is CN/MSS's standard for delivering IPA and APK files for mobile games. MSS developers will setup configs and repositories for games on Unity Cloud Build.

- The Android build system uses Gradle. Important jar files and dependencies are added in at build time by the Gradle script.
- UCB advanced options in the config will have a Pre-export method that sets the environment.
- Charles Proxy for Android requires the build be a Development Build.

2.4.3 Unity Architecture

- MediaMonks uses a combination of UniRx and Zenject open source plugins to make a reactive and event driven architecture.
- API event handlers execute commands with API data, and then other scripts subscribe to those commands.
- Searching the whole solution for those API events and finding the appropriate handler script is the only way to find the entry point of where API data is handled. Then searching the solution for the commands that subscribe to events will let you find the code that executes after an event.

2.4.4 Unity SDK's

- All SDK's have been modified from the MSS versions by MediaMonks, but are similar in that the same scripts do the same calls.
- FreewheelController script maintains the preroll ad calls. Search the solution for places that call preroll ads to find the exact spots the ads call.
- KochavaConfiguration script requests the attribution callback, and then KochavaHelper makes the Unity Analytics call.
- BannerOnLoad makes the sponsorship DFP call. InterstitialOnLoad makes an interstitial request. DFPManager and DFP script handle hitting the config with the game name as CN Arcade.
- Privacy and Terms SDK was heavily modified as well, but still hits the same config file with the CNPopupAgreement script. Displays the policy and terms in the UniWebView SDK.

- UniWebView constantly is coming out with updates on the Asset Store. There is plenty of documentation online on their website.

2.4.5 Unity Version Control

- Feature branches are called “feature/CNMO-####”
- Release branches are called “release/#.#” and should be based off develop branch.
- Develop branch is where all features are merged after a PR is approved.
- Master branch is only updated with release branches that are approved by QA.
- Typically, Unity PR’s are difficult to approve if they are scene changes or prefab changes. Try to keep files changed sizes to around 10 files, so that the PR process is easier on the reviewer. Don’t make massive PR’s.
- Unity Cloud Build configs have a branch option in Basic Info, so make sure to verify they are pointed to the release branch you want.

2.5 SDK Development

The Game/Stunt SDK is developed in Typescript, built using Webpack, and deployed as static JavaScript + CSS files to AWS. Games and Stunts link the hosted files in their respective index.html files.

- For example:
  - <link href=https://sdk.cartoonnetworkarcade.com/data/sdk/play.sdk.css>
    <scriptsrc=”htpps://sdk.cartoonnetworkarcade.com/data/sdk/play.sdk.js”></script>

There are two (2) major modules within the SDK: the Play module and UI module. The Play module corresponds to exposing API data to the Game/Stunt page. The UI module corresponds to the SDK overlay interface. Examples include, but or not limited to:

- back button,
- achievements overlay,
- score rank meter,
- collected figures overlay, etc.

The UI module retrieves data from the API and has no direct hooks for developers – it is read-only.Functions in the Play module are usable by game/stunt developers to access data for custom use. For example, in order to determine what figures should be displayed in CN Arcade if a developer needs to check which figures a user has unlocked in the app to determine what to display on the page, they can use the sdk.getUserFigures() function in the Play class. Behind the scenes, that function is making multiple API calls to consolidate the figure information needed for the client. New functions are added to the Play module when new data is required to be exposed to games/stunts that do not already exist.

3 ACR Ingestion Process

CN Arcade uses ACR to recognize content played from TV's and other video streaming devices/displays. This process allows CN Arcade to recognize audio from CN linear content being streamed through connected devices. Once CN content has been identified, the user can be rewarded CN Arcade prizes such as app figures or Cn Arcade currency.

MSS developed the application that handles processing of audio files via integration with ACR Cloud. ACR Cloud is a 3rd party company that provides the content recognition service. The content ingestion process is developed using Node.js. The architecture of this application is an AWS lambda service function which is triggered by an S3 event.

3.1 Git Repo location

The source code for the CNArcade ACR is stored in two version-controlled Git Repos. The main ACR application code can be found in the cn-acr-fingerprint GitRepo, but is dependent upon the recognizer codebase of the cn-recognizer GitRepo. Please see the ReadMe files contained at the root of the repos for implementation guidelines.

The project name and the GitRepo locations are as follows:

- Project Name: cn-acr-fingerprint
  - Fingerprint repository location: git@github.com:turnercode/cn-acr-fingerprint.git
  - Recognizer repository location: git@github.com:turnercode/cn-recognizer.git
    - The recognizer utility tool is needed for audio fingerprint verification.

3.2 AWS Cloud Resources

CN Arcade resources and services used to deliver application content are located in AWS Cloud. Below is a summary of these resources.

- Content delivery S3 bucket: cnplay-mediadrop
- Content archive S3 bucket: cnplay-acr
- DynamoDB table: cn-fingerprints-prod
- SQS queue: acr-notification-prod

3.3 Tools needed to process ACR content

Below are the external utility tools used by ACR Cloud to process content.

- cn-recognizer: Audio data sampling and content recognition utility.
- acrcloud_extr_linux: ACR Cloud utility tool to create audio/video fingerprint file.
- ffmpeg: Open source utility to manipulate audio/video media file.

3.4 ACR Automatic Content ingestion process diagram

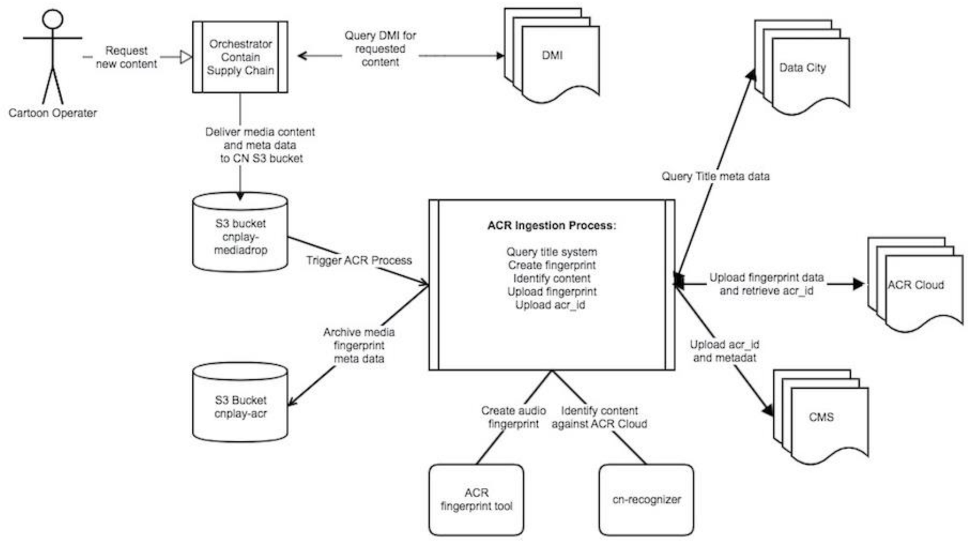


Figure 1: ACR Automatic Content Integration.

3.5 Application workflow

The ACR application monitors new content from the cnplay-mediadrop S3 bucket. Any Audio assets that are uploaded to this bucket is associated with metadata for the given asset. This will trigger the ACR ingestion process which uses the recognizer utility tool to sample the audio data stream. While sampling, calls are made upon the ACR cloud service to conduct content recognition of the new audio file.

If valid content is not found during this process, a new audio fingerprint file is created and uploaded to the ACR Cloud. The utility will retrieve the ACR ID from the ACR Cloud service, and upload this ID along with relevant metadata for this file to the Cn Arcade CMS for future use.

The audio data , meta JSON file, fingerprint, and log files are all moved to the cnplay-acr S3 bucket. A dynamo record is then created and saved for the process of the content.

3.6 ACR Contact information

The below provides contact information for all parties that are involved in the ACR ingestion process workflow.

3.6.1.1 ACR Cloud

ACR Cloud is a third-party vendor that provides content recognition service, client libraries, and tools. Any questions should be directed to:

Tony Li

Skype: astonysh1590

M: +86-18611325714

3.6.1.2 Orchestrator Content Supply Chain

WarnerMedia's Orchestator team created and supports the content delivery workflow. Any questions and/or issues should be directed to:

Joe Sciame

Slack: jsciame

3.6.1.3 Media Monks

3rd party application development company that partners with CN/MSS to develop Cn Arcade. Any questions and/or issues should be directed to:

Cris Kop - Senior Project Monk at MediaMonks

Email: cris.kop@mediamonks.com

phone: +31 6 20 41 27 23

3.6.1.4 MSS CN Developer

CN's dedicated MSS team created and supports the ACR content ingestion process. If you have any questions or issues please reach out to the following contact:

Contact: Yongli Cheng

Slack: yongli.cheng

4 CN Arcade Stunts

Stunts are static HTML webpages hosted remotely and displayed within a webview in CN Arcade. Stunts typically serve to supplement CN Arcade content and/or games with the primary goal of increasing user engagement. Many stunts direct users towards a specific action, like collecting a new figure or playing a new game. All stunts are accessed through the Home Feed in the app just like a game. Stunt HTML pages are able to access Arcade data through a JavaScript SDK. When integrated, the SDK exposes app/user data and methods for interacting with app content. For example, a developer can use the SDK to access information about figures a user has collected, unlock an achievement based on a button click, or deep link users to different parts of the app. In short, the SDK is the bridge between the mobile app and the HTML page

4.1 Development Environments

There are 3 development environments for CN Arcade:

- Production
- Acceptance
- Testing

Each environment has a corresponding CMS, database, API, and SDK. The SDK configuration used in a stunt html page will depend on the CMS environment used to create the stunt entity.

- For example:
  - https://sdk.cartoonnetworkarcade.com/data/sdk/play.sdk.js
  - https://sdk-acc.cartoonnetworkarcade.com/data/sdk/play.sdk.js
  - https://sdk-test.cartoonnetworkarcade.com/data/sdk/play.sdk.js

Each environment has a corresponding app build. Stunts created in the Production CMS will only be visible in the Production build of the app. In addition, those stunts will only have access to users/achievements/figures in the Production database; the same applies to the Acceptance and Testing environments.

For the purposes of this documentation, we will assume the stunt will be created in the production environment but remain in draft mode – meaning that the SDK will be able to utilize production data, but the stunt will only be visible to ‘draft users’ in the live app.

Contact Kartik Kini if you would like a new draft user account created for testing purposes.

4.2 Create Stunt entity in CMS

The first step in the process is to create the stunt entity in the CMS. If you are unsure if the stunt has already been created and don’t have CMS access, contact a CN producer.

- In the production CMS environment, click on Games & Stunts > Stunts > Add New
- Add a title and a placeholder ‘External Url’ (both can be changed later)
- If the stunt will have one or more associated achievements, you can add them in the ‘Achievements’ section at the bottom of the page.
    - **If you are unsure of the achievement requirements for the stunt, a CN producer can help you create these
    - Write down the achievement slug name(s) as you will need them later
- Finally, click the ‘Create’ button. However, if you are in the Production environment, do NOT select the option to ‘Publish’ at any given time.
- Write down the stunt UUID as you will need it later (Go to the list of all stunts, find your stunt, and click the grey info button on the far-right side of the row).
- If you have created one or more achievements, you must also go to the Achievements link in the side nav, search for your new achievement(s), and ‘Publish’ each.

4.3 Create a Test User

If you need a new ‘blank’ test user for troubleshooting stunt functionality, see the ‘User Accounts for Testing’ section of the SDK ReadME. Be sure to use the user token endpoint that corresponds with the desired environment:

- Production - https://api.cartoonnetworkarcade.com/api/user.token
- Acceptance - https://api-acc.cartoonnetworkarcade.com/api/user.token
- Testing - https://api-test.cartoonnetworkarcade.com/api/user.token

The developer will need to copy the user ‘id’ and ‘token’ UUIDs generated from the response for the next section.

4.4 Adding Scripts to Stunt Page

Be sure to include the SDK JS and CSS files in the index.html file of the stunt. See the SDK script/link tag section of the ReadME for details. Once again, be sure the URL is associated with the correct environment.

4.5 Configuring SDK for Local Development

In order to initialize the SDK outside of the app, you will have to add a hard-coded global object to the DOM which will provide the SDK with user data as well as point it to the correct stunt in the datavase. Here is an example of the object which needs to be added into the <head> of index.html:

```
<script>
  // Only set hardcoded configuration when *not* running in the app webview
  if (!window.navigator.userAgent.match(/CNPlayApp/)) {
    window['PlayConfig'] = {
          id: '683bc892-27a3-4867-9dc6-b25c223d8dd3',
          authToken: 'd5bb725f-4472-4f0a-b129-9aaa3df3982c',
          baseUrl: 'https://api.cartoonnetworkarcade.com',
          containerType: 'stunt',
          user: {
            id: '7f582962-94d5-4ce5-a760-e291d275542f',
            type: 'guest',
            displayName: 'Jane Doe'
          }
        };
      }
    </script>
```

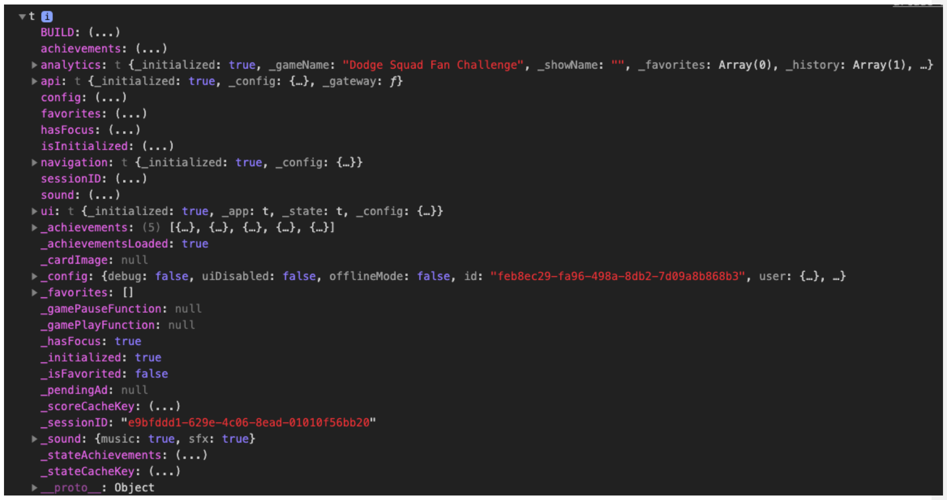
- The ‘id’ field should be populated with the stunt uuid you copied earlier from the CMS.
- The ‘authToken’ field should be populated with the user ‘token’ uuid copied from the response in the ‘Create a Test User’ section.
- The ‘user.id’ field should be populated with the user ‘id’ uuid from the ‘Create a Test User’ section.

4.6 Testing Initialization

The SDK will install a global variable that can be used in the stunt html or js files. You can add a function to the bottom of the page that calls this variable to determine if the SDK initialized properly:

```
<script>

  PlayPromise.then(sdk => {

    console.log('The SDK is ready, use it to do stuff');



    console.log(sdk);

  });

</script>
```

If the SDK initializes correctly, you should see the ‘sdk’ JSON object printed in the browser console when you open the stunt in the browser:



```
▼t i
   BUILD: (...)
   achievements: (...)
  ▶analytics: t {_initialized: true, _gameName: "Dodge Squad Fan Challenge", _showName: "", _favorites: Array(0), _history: Array(1), …}
  ▶api: t {_initialized: true, _config: {…}, _gateway: ƒ}
   config: (...)
   favorites: (...)
   hasFocus: (...)
   isInitialized: (...)
  ▶navigation: t {_initialized: true, _config: {…}}
   sessionID: (...)
   sound: (...)
  ▶ui: t {_initialized: true, _app: t, _state: t, _config: {…}}
  ▶_achievements: (5) [{…}, {…}, {…}, {…}, {…}]
   _achievementsLoaded: true
   _cardImage: null
  ▶_config: {debug: false, uiDisabled: false, offlineMode: false, id: "feb8ec29-fa96-498a-8db2-7d09a8b868b3", user: {…}, …}
  ▶_favorites: []
   _gamePauseFunction: null
   _gamePlayFunction: null
   _hasFocus: true
   _initialized: true
   _isFavorited: false
   _pendingAd: null
   _scoreCacheKey: (...)
   _sessionID: "e9bfddd1-629e-4c06-8ead-01010f56bb20"
  ▶_sound: {music: true, sfx: true}
   _stateAchievements: (...)
   _stateCacheKey: (...)
  ▶__proto__: Object
```

4.7 Troubleshooting Initialization

If you see errors in the console instead of that JSON object, please double check the following:

- Make sure all the UUIDs (user, stunt id, authtoken, etc) all correspond to the correct SDK environment.
- Make sure the manual window[‘PlayConfig’] object script is above the SDK js <script> tag in the document <head>.
- If the stunt is in ‘draft prod’, the user credentials in the window[‘PlayConfig’] object have to correspond with a user with draft prod access.
    - If you are using a draft prod user, add an additional namespace: ‘draft’ key/value to the window[‘PlayConfig’] object
- The SDK back button will only work in-app; clicking the button does nothing from the browser

Please contact an MSS developer if additional troubleshooting is needed.

4.8 Using SDK

The SDK provides access to information about the user and stunt itself through the PlayPromise object mentioned above. There are also many functions which can be called through the SDK. The SDK ReadME contains more details on additional functionality.

4.9 Example: Unlocking Achievement

The SDK exposes the ability to view user achievement progress and unlock achievements programmatically. However, you can only act on achievement(s) associated with the stunt entity in the CMS. The ReadME includes documentation on how to access achievement data.

Here is a quick example from that documentation:

```
window['PlayPromise'].then(sdk => {
  sdk.unlockAchievement('yummy-gummy').then(achievement => {
    console.log('Successfully unlocked achievement', achievement.name);
  });
});
```

For your purposes, you would replace ‘yummy-gummy’ with the achievement slug you copied from the ‘Create Stunt entity in CMS’ section mentioned earlier.

4.10 Pre-deployment Review and QA

It is general good practice for stakeholders to review the finalized stunt in the production app in draft mode. The goal is to ensure all functionality and all use cases are covered. It is also important to ensure that the page is responsive across all supported mobile and tablet devices. Important devices to test on include:

- iPhone 5/SE
- iPhone X
- iPhone 6/7/8
- Google Pixel 3
- Google Pixel 2 XL
- Nexus 6
- Galaxy Note 4/8
- Galaxy Tab
- iPad Mini
- iPad

- iPad Pro
- Google Pixel C

Also ensure all stunt functionality works on the latest versions of iOS (13) and Android (29).

4.11 Deploying a Stunt

Stunts created by CN MSS are currently hosted using the AWS S3 service. The cn-docroot bucket in the AWS ent-prod account uses a CDN that makes stunt webpages public behind this url: https://www.cartoonnetwork.com/...

A CN MSS developer can help you add your files to AWS if you do not already have a regular web hosting process. Please reach out to CN MSS if you need help.

Regardless of where the page is hosted, here is how you point the stunt entity in the CMS to your files:

- Copy the URL of your stunt page (i.e. https://www.cartoonnetwork.com/cn-arcade-stunts/figure-game/)
- Go to the stunt entity in the CMS you created earlier, edit, and paste it into the ‘External Url’ field.
- Click the ‘Update’ option but do NOT click on any option with ‘Publish’
- Ask a CN producer to create a Feed Item in the CMS for your stunt.
- You can now log in to the CN Arcade app (use a draft user account), find the feed item associated with your stunt, and open the page in the app.

Once you begin testing the stunt from within the app, ensure that the SDK back button works as well as any other custom SDK functionality (e.g. unlocking an achievement).

CN Producers/Stakeholders will ultimately determine when a stunt should be converted from draft prod to live. No action needs to be taken by the developer when a stunt is set to go "live" in the CMS

4.12 Source Code

All stunt pages are stored in version controlled GitHub repositories. The name convention is as follows:

- *cn-arcade_{brief stunt name}-stunt*
- Example: https://github.com/turnercode/cn-arcade_figure-game-stunt

Please update the master branch as often as possible and be sure it is up to date when the live hosted page(s) are updated. Also, please add any additional information to the ReadME that would be necessary for another outside developer to know about the stunt.

5 CN Arcade HTML5 Games

Please reference this section.

Commented [SJ1]: Need to reference the document/section that has already been written.

6 Work request and approval process

The following outlines the CN Arcade application process to estimate and complete operational and/ or maintenance development work outside of as well as CN's review/approval workflow.

6.1 Requesting work

In order to request work from the MSS CN development resources, CN will need to do one of two (2) things:

- Create a JIRA ticket with details on requirements,

OR

- Schedule a discovery meeting with MSS developer(s)/CN stakeholders to kick off a discussion regarding requested features and/or projects.

6.1.1 Creating a JIRA request

All work requested of the MSS development team should be submitted using the JIRA ticketing system which can be found here:

- https://jiraprod.turner.com/

Tickets should be assigned to the MSS resource manager, or can be directly assigned to an MSS developer.

CN Arcade JIRA ticket:

- CP - CN Arcade requests

JIRA tickets should include the following information:

- Goal(s) of requested project/feature/task,
- Requested LOE turnaround time,
- Expected review process (meeting(s), JIRA update(s), etc), and
- Any additional context needed to help scope request.

6.1.2 Discovery meeting

In the early phase of a project, CN may request a discovery meeting with MSS resources. The meeting is between the project team and development resources to understand the business goals and/or strategies of the project. Options regarding implementation will also be discussed at the meeting. A JIRA ticket is not needed to request a resource for this meeting.

6.1.3 Level of Effort (LOE)

The assigned MSS Developer should provide an LOE document for any projects or new feature requests. The LOE template should be used for this information and can be found here:

- https://docsprod.turner.com/display/MSSLOE/MSSLOE

Please see section 3.1.4 for information on submitting LOE for review.

6.1.4 LOE review and CN producer approval

Once the LOE has been created, it must be attached to the corresponding JIRA ticket. The assigned CN producer requesting work will review the LOE and provide feedback. The producer will also confirm that the developer recommended approach and estimate corresponds to the original request. The goal of the LOE process is to confirm that all CN requirements are finalized and approved before development is started.

6.1.5 MSS QA Verification

All CN Arcade changes must be verified by CN and/or MSS QA before changes can be deployed to production. An MSS Developer will work with appropriate QA team to ensure JIRA ticket and/or feature requirements are clear and a testing strategy has been identified. Prior to production deployment QA will verify and approve all bug fixes and/or new features in a low (development/staging) environment. Target date for production deployments is every Monday, but this date is subject to change based on urgency of request.

- QA will update the JIRA ticket with the following status once testing completed:
    - Ready for Deployment - verified and approved bug fixes / new features
    - Open - non-approved tickets and assigned back to the PM or a developer
- On day of deployment, immediately after deployment is complete, QA executes automated “Smoke Test” scripts to ensure stability of supported platforms in production.

6.1.6 Stakeholder approval and resource assignments

CN and MSS Stakeholders attend a bi-weekly prioritization meeting to review approved LOEs and confirm resource assignments. LOEs in review and resource assignments can be found at the following location:

- https://docsprod.turner.com/pages/viewpage.action?pageId=40968241

6.1.7 CN approval for production deployment

CN must approve all deployments impacting production via email. The email shall include details on a comparison between environments. This ensures that CN can fully understand the impact of the proposed changes.

The email shall be sent to the following:

- CN: Carrie, Sherri, Blaine, Jeff, Terri
- MSS: Branden, Yongli

6.1.8 Change Management process for production changes

MSS has defined a change management and approval process that all MSS developers must follow for production changes. Please note that all CN stakeholders are part of the communication distribution list for any and all changes submitted via the pre-approved change management template.

The steps needed to submit the MSS change management request form are as follows:

- Go to: https://warnermedia.service-now.com/navpage.do,
- Select “Change”,
- Select “Create New”,
- Select “Standard: Select from available pre-approved change templates”,
- Select “MSS”,
- Select “Application, website or infrastructure,
- Answer change management form questions.

7 System Integrations

The CN Video application integrates with a number of shared and dedicated systems. The section below provides a high-level description of these systems and contact information for system owners.

System	Description	Contact	Team Slack Channel
Zendesk	Customer Service management tool	Leah Randall	
Conviva	Video Analytics Product (Analytics)	Kenny Tyler	

DFP	Display ad server	Benjamin Rasmussen	#cn-ads
Ensighten	App Analytics API endpoints	Kenny Tyler	
FreeWheel	Digital Ad Server – (Ads Management)	Benjamin Rasmussen	#cn-ads

Figure 2: High level description of the systems that integrates with the CN Arcade application.

8 Monitoring

All CN products in production are monitored by WarnerMedia's DOC. The DOC works with brands to provide first response issue resolution. The DOC engages all parties that are needed for major incidents and facilitates communication.

8.1 Emergency response procedure

DOC liaison, MSS, and CN producers work together to create an emergency response plan for all CN products in production. This plan outlines the DOC steps to triage a production issue. CN Arcade ERP can be found here:

- https://docsprod.turner.com/pages/viewpage.action?pageId=133473087

8.2 Big Panda

The DOC uses a product called Big Panda to consolidate alerts from various systems. When a new product and/or feature is created, MSS developers and DOC will work together to ensure Big Panda tracking has been implemented.

The following is a list of all CN Big Panda alerts including CN Arcade.

- https://docsprod.turner.com/display/CARTOONNETWORK/CN+Alerts+available+in+Big+Panda

The following is a complete is a list of systems that integrate with Big Panda

- https://docsprod.turner.com/display/INFRASVCS/BigPanda+Integrations

8.3 Production escalation process

Production issues may need to be escalated to the DOC for assistance. The DOC monitors the following slack channel:

- #doc-incident-bridge.

The reporter must provide the DOC details of the prod issue, user impact, device details, and any other relevant information to facilitate the support team’s understanding and engagement in resolving the production issue.

9 URL LInks

This section will serve as a quick reference guide for URL links found throughout the document.

9.1 API Git Repo

- https://github.com/turnercode/cartoon-play-api

9.2 CN Arcade API Domain URLs:

- Test: https://api-test.cartoonnetworkarcade.com/api/{apiPath+}
- Acceptance: https://api-acc.cartoonnetworkarcade.com/api/{apiPath+}
- Production: https://api.cartoonnetworkarcade.com/api/{apiPath+}

9.3 Jenkins Instances

- Production
  - https://cartoon-network-cn-play-app-prod.sysops.us-east-1.321061860761.monkapps.com/jenkins

- Acceptance
  - http://cartoon-network-cn-play-app-acc.sysops.us-east-1.321061860761.monkapps.com/jenkins

- Testing
  - http://cartoon-network-cn-play-app-test.sysops.us-east-1.321061860761.monkapps.com/jenkins

9.4 Setting up local CMS environments

- https://github.com/turnercode/cartoon-play-cms

9.5 CMS Acceptance

- https://cms-acc-l3d9h0i366.cartoonnetworkarcade.com/

9.6 CMS Production

- https://cms-p5z8zbvsz0.cartoonnetworkarcade.com/dn7bUiCG8v3y5ry9/login

9.7 CMS Producer information

- https://docsprod.turner.com/pages/viewpage.action?spaceKey=CPPA&title=CMS+Information

9.8 Create a Test User

- Production - https://api.cartoonnetworkarcade.com/api/user.token
- Acceptance - https://apiacc.cartoonnetworkarcade.com/api/user.token
- Testing - https://api-test.cartoonnetworkarcade.com/api/user.token

9.9 Creating a JIRA request ticket

- https://jiraprod.turner.com/

9.10 Level of Effort (LOE) Template

- https://docsprod.turner.com/display/MSSLOE/MSSLOE

9.11 LOE's in Review and Resource Assignments

- https://docsprod.turner.com/pages/viewpage.action?pageId=40968241

9.12 Change Management Request Form

- Go to: https://warnermedia.service-now.com/navpage.do,

9.13 CN Emergency Response Procedure

- https://docsprod.turner.com/pages/viewpage.action?pageId=133473087

9.14 CN Big Panda Alerts List

- https://docsprod.turner.com/display/CARTOONNETWORK/CN+Alerts+available+in+Big+Panda

9.15 Systems that Integrate with Big Panda

- https://docsprod.turner.com/display/INFRASVCS/BigPanda+Integrations