AGOL and Portal Backup Script Read Me

This document is for understanding how the AGOL and Portal Backup Script works. The script iterates through a list of groups on AGOL and Portal and downloads a JSON copy of every Feature Service shared within the group. It runs a request on the service querying for records where "OBJECTID IS NOT NULL" and thus requires the feature service to have an OBJECTID field. The JSON can be extracted using the Extract JSON tool in the Yosemite Custom Tools Enterprise Data Management Toolbox

# Qualifying Data

Must be a Feature Service and have an OBJECTID Field. Datasets that are edited frequently or on unstable platforms (cough Portal cough cough) are good candidates. Project data that may not end up in an SDE but is important for ongoing Park operations.

# Files and Infrastructure

## In AGOL\_PORTAL\_BackupScript folder

* DownloadFromAGOL.py: the python script that contains the code that does all the work
* Settings\_download\_agol.yaml: the configuration file
  + AGOL\_SOURCE / PORTAL\_SOURCE: information about AGOL
    - Username: headless account username
    - Password: headless account password
    - Token URL: where to get the token requests uses to run queries
    - Referer: important for the token
    - GIS\_URL: the url used to access the platform – [https://nps.maps.arcgis.com](https://nps.maps.arcgis.com/) or <https://gisportal.nps.gov/portal>
  + GROUP\_CSV: the list of groups from which the data needs downloading
  + LOG\_FOLDER: location where to write log files when script runs nightly,
  + DOWNLOAD\_FOLDER: where to download JSON zip files
* DownloadGroups.csv: a csv containing information on all the groups to download including
  + PLATFORM: which platform the group is on – AGOL or Portal
  + GROUPNAME: for display and script write out purposes, makes it easy to read the csv
  + GROUPID: the unique id for the group on AGOL or Portal, used by the script to identify and get content from the group
* Shortcut to backup location

## In AGOL\_PORTAL\_Backups folder

* Three folders: AGOL, LOGS, and PORTAL
* The script's logs are written to the LOG folder
* AGOL and PORTAL are where the JSONs end up
  + Within each are folders for each group backed up by the script
  + If a specified group does not have a folder the script will create one
  + Permissions can and should be set on these folders to prevent tampering and ensure security of sensitive data

## AGOL and PORTAL Groups

* Created or designated by GIS Admin
* See "How to Designate or Create a Backup Group" section below

# Detailed Code Breakdown

* Main function
  + Reads the YAML File
  + Starts logging the process
  + Runs through AGOL backup
  + Runs through PORTAL backup
* Run function
  + Reads the CSV and identifies all groups on the specified platform
  + Makes a [GIS object](https://developers.arcgis.com/python/api-reference/arcgis.gis.toc.html) to interact with the platform
  + Gets a token to make query requests to feature services
  + Iterates through list of groups...
    - Identifies all items in group (flags when there are no feature services in a group in log)
    - Identifies JSON download folder ([platform]/[group name]/[platform]\_download\_[yyyymmdd]\_[hhmm]
    - Iterates through the items in the group and [queries](https://developers.arcgis.com/rest/services-reference/enterprise/query-feature-service-.htm) them for OBJECTID IS NOT NULL records, dumps the resulting json text into a .json file
    - Zips the JSON download folder up, deletes the old folder

# How to Designate or Create a Backup Group

* Talk to the data stewards
* Create or identify group on AGOL or Portal
* General backup group naming convention: YOSE [Workgroup Name] Data Backup
* Recommend using a consistent thumbnail
* Add platform, group name, and group ID information to DownloadGroups.csv

# How to Extract JSON

Follow [instructions in EGIS user Manual](http://gis.yose.nps.gov/help/user/AGOLAndPortal/EGIS_JSONBackup.html)

# Lessons Learned / Common Problems / Troubleshooting / Limitations

* JSONToFeature Tool
  + Very finnicky, does not accept JSON as they are downloaded from AGOL or Portal
  + The custom Extract JSON tool removes information about fields which gets the JSONTo Feature Tool to work
* Global IDs
  + JSONToFeature Tool is messing up global id imports, they came in as {0000000000}
  + Script removes global id field identification
  + Script converts global id field to string field in JSON, thus importing global ids as strings
  + Might have something to do with gdb needing global ids in ALL CAPS surrounded by squigglies but agol global ids are generally lowercase with no squigglies {}