Config Validator

Validate configuration as part of your CI process

Config Validator

- Written in Swift and runs on macOS.
- Validates JSON and / or Property Lists (.plist).
- Supports upload to AWS S3.
- Supports AWS CloudFront cache invalidation.
- Outputs to console & Slack.

Usage

- `--files` [file1 file2 file3]
- `--upload-method awss3`
- `--upload-urls` [s3://bucket/file1 s3://bucket/file2 s3://bucket/file3]
- [Optional] `--cloudfront-distribution-id` <identifier>
- [Optional] `--silent`
- [Optional] `--verbose`

Validation

- Detects whether files are JSON or Property List format and checks that these contain valid content for the file format.
- JSONSerialization in the Foundation framework for macOS used to check JSON validity.
- PLUtil used to check Property List validity.

From the macOS man page:

```
plutil can be used to check the syntax of property list
files, or convert a plist file from one format to
another.
```

Git

- Check whether any of the validation files have been modified in the latest commit.
- If using **shallow clone** min depth of two should be specified otherwise the difference between the last two commits cannot be determined.

```
git clone --depth 2
```

- Files are always validated even not modified in latest commit.
- By default, files are only uploaded if they have been modified in latest commit.
  - `--force-upload` parameter can be used to skip this check.

Uploading

- Upload to AWS S3 currently the only supported upload destination.
- Indicate intention to upload to S3 using `--upload-method awss3` parameter.
- S3 bucket is determined from the s3 URL passed to the `--upload-urls` argument.
- `--files` arguments are one-to-one mapped to `--upload-urls` respectively.
  - Upload will not proceed unless the number of arguments is equal.

AWS CLI

- AWS Command Line Interface
  - Used to optionally upload to S3
  - Used to optionally invalidate CloudFront cache
- Credentials are read from a credentials file in the user's home directory: ~/.aws/credentials
  - Credentials are never passed to Config Validator itself.

AWS CLI Credentials

- Configuring the AWS CLI:

https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-getting-started.html

- Configure command:
  - **`aws configure`**
- ~/.aws/credentials:

```
[default]

aws_access_key_id=<aws access key id value>

aws_secret_access_key=<aws secret key value>
```

IAM Permissions

- Use AWS Identity and Access Management (IAM) to restrict access to resources in your AWS account.
- Create a separate IAM user for use by your build server as part of your CI process.
- Assign an IAM policy which restricts access to resources to this user.
  - Can be defined in JSON or through the Visual Editor in the IAM section of the AWS console.

IAM Permissions

- Upload to AWS S3 requires the `s3:PutObjects3:PutObject` permission.
- Setting files to public in AWS S3 requires the `s3:PutObjectAcl` permission.
- Cache invalidation in CloudFront requires the `cloudfront:CreateInvalidation` permision.

Cache Invalidation

- Config Validator is able to invalidate CDN cache
- Only currently supported CDN is AWS CloudFront
- If `--cloudfront-distribution-id` parameter passed then all files successfully uploaded to S3 will be made public.
- If `--cloudfront-distribution-id` parameter passed then all files successfully uploaded to S3 will be invalidated in CloudFront.

Slack

- Configure an incoming web hook
- Generates a unique URL allowing you to POST a JSON payload containing your message.
- Pass the webhook URL to IPA Uploader using the optional `--slack-url` parameter.
- Once specified all output will be sent to both the console and Slack.

Messaging Levels

- `–silent (–s)` prevents output being emitted to console or Slack.
- `––verbose` (`–v`) will cause additional diagnostic information to be emitted.

Config Validator

https://github.com/rwbutler/ConfigValidator

- Build from the Xcode project.
- Download the binary from the repository.

Building

- Download & install Xcode.
- Open the Xcode project (.xcodeproj).
- Select the `config-validator` scheme to compile with optimizations.
- Product -> Archive to build.

References

- Amazon - AWS Command Line Interface

https://aws.amazon.com/cli/

- Amazon - AWS Identity and Access Management

https://aws.amazon.com/iam/

- Apple - JSONSerialization

https://developer.apple.com/documentation/foundation/jsonserialization

- Slack - Incoming Webhooks

https://api.slack.com/incoming-webhooks