Taming SwiftLint in CI

Sebastian Hagedorn
CocoaHeads Aachen, 29.11.2018
Taming SwiftLint in CI

From https://github.com/realm/SwiftLint:

```sh
/bin/sh

if which swiftlint >/dev/null; then
    swiftlint
else
    echo "warning: SwiftLint not installed, download from https://github.com/realm/SwiftLint"
fi
```

- Show environment variables in build log
- Run script only when installing
1 Sharing a Linter Configuration

- **Goal**: One config for all projects
  - Discuss style once
  - Share code between projects without changes

- **Challenge**: Config has to live in project root directory (where `swiftlint` is called from)
1 Sharing a Linter Configuration

- **App A**
  - default config

- **Shared Tooling**
  - copy-config.sh
  - default config

- **App B**
- **App C**
- **App D**

...called from Xcode build phase
1.1 Sharing a Linter Configuration

App A
- merged config

…called from Xcode build phase with local config

Shared Tooling
- merge-config.sh
- default config

git submodule

App B

App C

App D
2 Calling swiftlint

- Linter has to be called from project root directory (where config file lives)
- Different output format for local Xcode builds and Jenkins (checkstyle XML)

```bash
1 if [ ! -z "${JENKINS_URL+x}" ]; then
2   echo "SwiftLint: Jenkins build, skipping linter."
3   exit 0
4 else
5   swiftlint
6 fi
```
3 Adding a Bouncer

Start → Checkout → Verify → Archive → End

- Build Alpha
- Build App Store
- Build SDK
- Run Tests
- SwiftLint
3 Adding a Bouncer
4 Pinning Down the SwiftLint Version

- **Goal**: Consistent reporting in local and CI builds

- Make use of new rules and enforce them consistently
  - Older SwiftLint versions ignore unknown rules
  - Newer versions may include fixes that surface previously ignored issues

- Actual `swiftlint` call is wrapped in (shared) script that enforces SwiftLint version
  - Still needs to be called from project root
  - Script forwards all arguments to `swiftlint` invocation
  - Don't `echo` to stdout: Reserved for linter output
4 Pinning Down the SwiftLint Version

```bash
if ! [ -x "$(command -v swiftlint)" ]; then
    echo "error: SwiftLint not installed, download from https://github.com/realm/SwiftLint"
    return 4
fi

# minimum swiftlint version:
# script will fail if version is lower than this
LINTER_VERSION_MIN='0.27'
INSTALLED_LINTER_VERSION=$(swiftlint version)

printf '%s
' ${LINTER_VERSION_MIN} ${INSTALLED_LINTER_VERSION} | sort --version-sort --check

echo "Running swiftlint from ${PWD}." 1>&2
swiftlint "@"
```
5 Pinning Down the Xcode Version

- **Issue**: Remaining reporting inconsistencies

- SwiftLint relies on current Xcode installation (as per `xcode-select`) and its Swift libraries
- Newer SwiftLint versions may require certain minimum Swift lib versions

- **Solution**:
  - Pin down Xcode version in shared linter invocation script by setting/exporting `DEVELOPER_DIR`
  - Move linter invocation to separate target (instead of build phase)
    - Xcode version for linting not necessarily the Xcode version that builds the app
5 Pinning Down the Xcode Version

```bash
1 # ... linter version pinning up here ...
2
3 BASE=$(cd "$(dirname "$0")" && pwd)
4 XCODE_VERSION=10.0
5
6 . "${BASE}/xcode-enforce.sh" 1>&2
7 xcode_enforce ${XCODE_VERSION} 1>&2
8
9 echo "Running swiftlint from ${PWD}." 1>&2
10 swiftlint "$@
```
5 Pinning Down the Xcode Version

External Build Tool Configuration

- Build Tool: ./script/prepare-lint.sh
- Arguments: -performLocalLinting
- Directory: ../..

Pass build settings in enviro...
6 Future Improvements

- Allow proper merging of local/shared configurations
  - We currently only append files, assuming they are mutually exclusive
  - Will allow us to provide custom per-project rules
Thanks for your attention.

Sebastian Hagedorn
iOS Lead Developer

- @hagidd
- sebastian.hagedorn@sumup.com