

Analyzing code coverage at method level.

1. Set up Dev Environment in Local

- a. `git clone https://github.com/viacomcbs/roku-vmn-core.git`
 - b. `curl -sSL https://get.rvm.io | bash`
 - c. `source /Users/2185851/.rvm/scripts/rvm`
 - d. `export CONFIGURE_ARGS=""`
- for ext in openssl readline libyaml zlib; do
- `CONFIGURE_ARGS="${CONFIGURE_ARGS} --with-$ext-dir=$(brew --prefix $ext)"`
- done
- `rvm install 3.3.2`
- e. `rvm reinstall ruby-3.1.6 --with-openssl-dir=/opt/homebrew/Cellar/openssl@3`
 - f. `gem install bundle`
 - g. `brew install shared-mime-info`
 - h. `bundle install`
 - i. `roku --configure(generates ~/.roku_config.json)`
 - j. Update `'.roku_config.json'` like below

```
1  {
2      "devices": {
3          "default": "RokuUltra",
4          "RokuUltra": {
5              "ip": "192.168.2.2",
6              "user": "rokudev",
7              "password": "via123"
8          }
9      },
10     "projects": {
11         "default": "core2"
12     }
13 }
```

2. Then replace this file... Refer to these notes to run the code coverage.

Need to replace the folder in this path with this unzipped file `brands/vmn-player/UVPRoku`

3. Run this command “-ls vh1-staging“

4. Run this command to start the server which instruments the code

```
roku --coverage --stage vh1-coverage-testing --coverage-file ./coverage.json
```

5. CTRL+C to stop and a JSON file gets generated

Reference: <https://github.com/viacomcbs/roku-vmn-core/pull/2363>

Test Steps

- combined method:

- run the command `roku --coverage --stage vh1-coverage-testing --coverage-file ./coverage.json`
 - this will launch an app on your roku with a special coverage build
 - it will also start a tcp socket server on your local computer and wait for the app to connect to it
 - when the app connects to the tcp socket it will send data to the server describing the files that were loaded
 - the app will then continue to send coverage information throughout the duration of the app session.
 - press `ctrl-c` while the server is running and it will send the session. If the app still has data to send it will wait until it is done.
 - You can press `ctrl-c` again to stop waiting for data and generate the report right away.

- separate method:

- run `roku --coverage-build --stage vh1-coverage-testing`
 - this will generate a zip file that can be side loaded on to a device.
 - run `roku --coverage-server --coverage-file ./coverage.json`
 - this will launch the tcp server on your local machine.
 - side load the zip file generated from the first command.
 - follow the steps as above in the combined method.