Developer Onboarding

Prerequisites

To develop, build and test Firefly you need to install the following tools and software on your local computer or cloud desktop:

1. Install the macOS and Linux package manager Homebrew if it is not already on your machine.
2. Install nvm. Homebrew is a recommended method:

```
brew install nvm
```

3. Install the latest version of node with the following command - the minimum Node.js version required is 16.

```
nvm install node
```

4. Follow steps to install the latest version of npm with a method you choose.
5. Next, install the package manager yarn using npm:

```
npm install --global yarn
```

6. Follow steps to install the AWS CLI V2 2.13.16 or newer.
7. Verify your AWS CLI installation:

```
which aws
```

8. Next, set up your Amazon Developer Account (ADA) CLI:
    i. You must be able to authenticate with Midway for the next steps.

ii. Builder Toolbox is required to install the ADA CLI. Follow the Toolbox instructions at BuilderHub to install.

iii. To install the ADA CLI, run:

```
mwinit
```

and

```
toolbox install ada
```

9. Make sure you have access to this bindle.

- Sign in and select **See all permissions** to check for your name on the list.
  This bindle provides access to our IAM role.

10. Configure Commitizen on your local development desktop. You can follow the steps as described in the Configure Commitizen.

Check out FireFly code

1. Create a development directory on your computer. In a script later in this document we will reference a hypothetical directory named `~/Code/Firefly/`
2. Authenticate using Midway:

```
mwinit
```

Note: Running mwinit means no additional SSH setup is necessary. Your Midway session will last 24 hours.

3. Clone the Firefly package to your computer:

```
git clone git@ssh.gitlab.aws.dev:amazonmusic/musicfirefly/amu_webapi
```

Retrieve a new accessKey and secretKey for your AWS credentials

To get AWS credentials to develop in FireFly, change directory to amu_webapi and run:

```
yarn authDev firefly-dev
```

Note: The keys will be valid for one hour, so you may need to re-run when deploying later. This will automatically overwrite the values under this profile in your ~/.aws/credentials file:

```
[firefly-dev]
aws_access_key_id=AQCDA....
aws_secret_access_key=bcQq25....
aws_session_token=IQoJb....
```

Tip: If you encounter issues with your profile, you may need to delete it from `~/.aws/credentials` and re-generate it with `yarn authDev firefly-dev`.

Log in to CodeArtifact

Now that your profile is configured, you need to access the CodeArtifact repository to get all the necessary npm packages for development.

First, run this command to attach an auth header to all requests to get packages from CodeArtifact. This will prevent attempts without using authentication.

```
echo 'always-auth=true' >> ~/.npmrc
```

Next, log into the FireFly CodeArtifact repository via the following command:

```
AWS_PROFILE=firefly-dev aws codeartifact login --tool npm --domain amazon
--domain-owner 149122183214 --repository music-sonic --region us-west-2
```

Automate your FireFly login

Next, automate your Firefly login with a script in `~/.zshrc` or `~/.bash_profile`

In the example script below,

- Replace INSERT_ALIAS with your Amazon alias
- Replace the folder path with the one in your own computer. For example, (~/Code/Firefly/amu_webapi)

```
firefly() {
    cd ~/Code/Firefly/amu_webapi
    yarn authDev firefly-dev
    export AWS_PROFILE=firefly-dev
    aws codeartifact login --tool npm --domain amazon --domain-owner
149122183214 --repository music-sonic --region us-west-2
    export USER=INSERT_ALIAS
    export STAGE=dev
    export NODE_OPTIONS=--max_old_space_size=8172
}
```

Configure Commitizen

It is required for all contributors to install Commitizen and issue commits using this command line tool because this package and all packages under the Amazon Music FireFly platform make heavy use of semantic-release for automating deployments and version management.

1. Run

```
yarn global add commitizen
```

2. If `cz-conventional-changelog` is not installed as a dependency of Commitizen, run

```
yarn global add cz-conventional-changelog
```

3. Next, run

```
echo '{ "path": "cz-conventional-changelog" }' > ~/.czrc
```

4. To issue a commit with Commitizen, run:

```
git cz
```

Tip: If after running `git cz` you get the error `git: 'cz' is not a git command` or `cz is not command`, the solution is to get the yarn global bin path by running `yarn global bin` and then add that path in path variable in the `~/.zshrc file`.

- One method is to use the following command to add a directory to the PATH: `export PATH="/path/to/your/directory:$PATH"`
- Verify that the directory has been added: `echo $path`

When you run `git cz` you will receive a prompt for styling and adding metadata:

```
➜  ng-poopy  master ✗  git add .
➜  ng-poopy  master ✗  git cz

All commit message lines will be cropped at 100 characters.

? Select the type of change that you're committing: (Use arrow keys)
❯ feat:     A new feature
  fix:      A bug fix
  docs:     Documentation only changes
  style:    Changes that do not affect the meaning of the code
            (white-space, formatting, missing semi-colons, etc)
  refactor: A code change that neither fixes a bug or adds a feature
  perf:     A code change that improves performance
  test:     Adding missing tests
  chore:    Changes to the build process or auxiliary tools
            and libraries such as documentation generation
```

Next Steps

- Choose to develop locally (recommended) or deploy in a devstack region
- See the Postman and Insomnia setup guides.