WebFX Documentation

Table of Contents

Introduction

Getting started

Prerequisite

To develop WebFX applications, you will need the following software installed on your development machine:

- JDK 13 or above
- Maven
- Git
- Your preferred Java IDE

Be sure that you can invoke `java`, `mvn` and `git` directly from the terminal. The WebFX CLI will invoke `mvn` and `java` without specifying their full path.

Introducing the WebFX CLI

The WebFX CLI is an essential tool to assist you developing WebFX applications. It will create and maintain your modules and build files, such as:

- pom.xml (Maven module files)
- module-info.java (Java module files)
- module.gwt.xml (GWT module files)
- and some other files required for a successful compilation

```
Failed to generate image: Could not find the 'dot' executable in PATH; add it to the
PATH or specify its location using the 'graphvizdot' document attribute
digraph g {
    "webfx.xml" -> "webfx-cli"
    "Your Java source" -> "webfx-cli"
    "webfx-cli" -> "pom.xml"
    "webfx-cli" -> "module-info.java"
    "webfx-cli" -> "GWT module.gwt.xml
super sources, bundles, etc..."
    "webfx-cli" -> "GraalVM conf"
}
```

Your only inputs will be centralized in the WebFX module files (named webfx.xml), and the tool will automatically generate the rest of the build chain from these webfx.xml files, combined with an analysis of your source code. This includes all your dependencies. The tool also takes over the aspects of cross-platform development: when a feature is platform-dependent (a different implementations exists for each platform), the tool will pick up the right modules in the final executable modules (the ones matching the platform targeted by the executable module).

Installing the WebFX CLI

As we haven't published any release at this stage yet, the way to install the WebFX CLI for now is to clone the webfx-cli repository, and build it with Maven.

We will distribute the WebFX CLI in a better way later, when we will publish the first official release.

Cloning the webfx-cli repository

HTTPS

```
git clone https://github.com/webfx-project/webfx-cli.git
```

SSH

```
git clone git@github.com:webfx-project/webfx-cli.git
```

Building webfx-cli with Maven

This is achieved by running the Maven *package* goal under the webfx-cli directory:

```
cd webfx-cli
mvn package
```

As previously mentioned, WebFX CLI requires JDK 13 or above (you will get a build error with lower versions).

This generates an executable fat jar in the target folder that we can execute with java:

```
java -jar target/webfx-cli-0.1.0-SNAPSHOT-fat.jar --help
```

Creating a permanent *webfx* alias

To easily invoke the WebFX CLI from a terminal, we need to create a permanent *webfx* alias to it. This is done with the following command (to run under the webfx-cli directory):

Linux

```
echo "alias webfx='java -jar $(cd "$(dirname "$1")" && pwd -P)/$(basename
"$1")/target/webfx-cli-0.1.0-SNAPSHOT-fat.jar'" >> ~/.bashrc ①

source ~/.bashrc ②
```

① Adding the alias to the shell profile

② Applying it to the current session

macOS >= Catalina

```
echo "alias webfx='java -jar $(cd "$(dirname "$1")" && pwd -P)/$(basename
"$1")/target/webfx-cli-0.1.0-SNAPSHOT-fat.jar'" >> ~/.zshrc ①

source ~/.zshrc ②
```

① Adding the alias to the shell profile

② Applying it to the current session

macOS < Catalina

```
echo "alias webfx='java -jar $(cd "$(dirname "$1")" && pwd -P)/$(basename
"$1")/target/webfx-cli-0.1.0-SNAPSHOT-fat.jar'" >> ~/.bash_profile ①

source ~/.bash_profile ②
```

① Adding the alias to the shell profile

② Applying it to the current session

Windows PowerShell

```
If (!(Test-Path $profile)) { New-Item -Path $profile -Force } ①

"`r`nfunction webfx([String[]] [Parameter(ValueFromRemainingArguments)] `$params) {
java -jar $((Get-Item .).fullName)\target\webfx-cli-0.1.0-SNAPSHOT-fat.jar `$params
}`r`n" >> $profile ②

If ($(Get-ExecutionPolicy) -eq "Restricted") { Start-Process powershell -Verb runAs
"Set-ExecutionPolicy -ExecutionPolicy RemoteSigned" -Wait } ③

. $profile ④
```

① Creating a PowerShell profile if it doesn't exist

② Adding the alias (implemented as a function) to it

③ Lowering the execution policy if necessary to execute the profile

④ Applying it to the current session

Then you should be able to invoke the CLI tool from the terminal:

```
webfx --help
```

Updating the WebFX CLI to the latest version

If later you want to update the WebFX CLI to the latest SNAPSHOT version, you just need to update your local repository and rebuild it with Maven. This is done through the following commands (under your webfx-cli local folder):

```
git pull
mvn package
```

Creating your first WebFX app

Creating and initializing your repository

Let's create our first WebFX application. We need to create the repository folder and ask the WebFX CLI to initialize it, passing it the groupId, artifactId and version of our application.

```
mkdir webfx-example
cd webfx-example
webfx init org.example webfx-example 1.0.0-SNAPSHOT
```

Creating your application modules

```
webfx create application --class org.example.webfxexample.WebFxExampleApplication
--helloWorld
```

Building your application

```
webfx build
```

Running your application

You can run the OpenJFX version of your application with the following command:

```
webfx -m webfx-example-application-openjfx run
```

You can run the GWT version of your application with the following command:

```
webfx -m webfx-example-application-gwt run
```

Developing in your IDE

We will give the instructions for IntelliJ IDEA, but you should be able to easily transpose them to other Java IDEs such as NetBeans or Eclipse.

Opening the project

Configuring the OpenJFX application

Building and running the GWT application

Making changes

```
webfx update
```