iPlug2 containerized WAM development

Oli Larkin, November 2020

These instructions describe how to do “containerized development” with iPlug2 and use github’s cloud services to automate builds. We can edit the C++ code for an iPlug2 project inside a container where we can build a Web Audio Module using the emscripten toolchain, which is preinstalled. We run a script to build and host the WAM in the container using a local web server and the ports are forwarded so that we can access the web server in a browser on our machine. When we are happy with the changes to our C++ code, we commit the changes using git version control, and push those changes to github. Using the included CI/CD[1] scripts github’s servers will then publish our WAM to a github pages website (e.g. https://iplug2.github.io/iPlug2OOS). There are also CI/CD scripts to build native VST3 and Audiounit versions of the iPlug2 project which will be published to the “releases page” of the github repository.

Throughout this document when you see **PROJECT_NAME**, that should be replaced with whatever you want to call your project, e.g. **MySynthesiser**. It should be a single word, no spaces, and not beginning with a number.

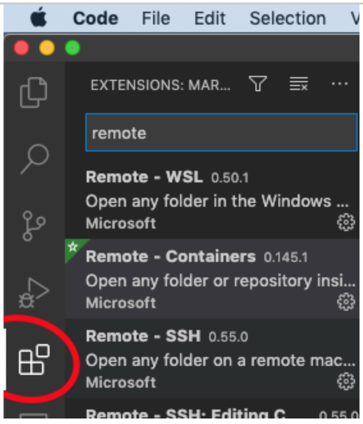
Prerequisites:

Please install the following software. You should have:

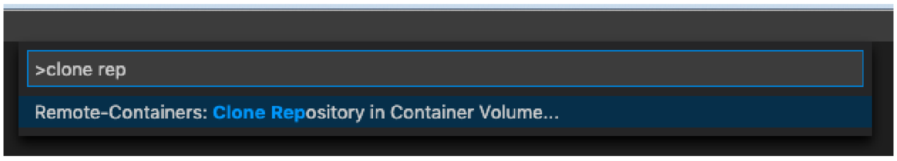
- A Mac, Windows or Linux machine
- A github account https://github.com
- Visual Studio Code https://code.visualstudio.com
- Docker Desktop https://www.docker.com/products/docker-desktop
- A Web Browser (preferably Chrome or Firefox)

Steps:

1. Use the iPlug2OOS template repository to create a new github repo
   1.1. Visit https://github.com/iPlug2/iPlug2OOS
   1.2. Click the **Use this template** button and input a new name for your project. This will be the name that is used in your WAM URL when published by github pages, e.g. https://YOUR_GITHUB_USERNAME/**PROJECT_NAME**.github.io . In the following examples the new repo is also called iPlug2OOS, but I suggest changing it to the name of your project.


Code File Edit Selection
EXTENSIONS: MAR...
remote
Remote - WSL 0.50.1
Open any folder in the Windows ...
Microsoft
Remote - Containers 0.145.1
Open any folder or repository insi...
Microsoft
Remote - SSH 0.55.0
Open any folder on a remote mac...
Microsoft

3. Create the container
    3.1. Open the VSCode “Command Palette” and run the command called **“remote-containers: clone repository in container volume”**



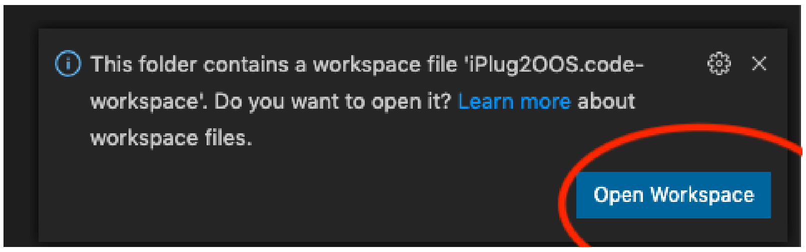
```
>clone rep
Remote-Containers: Clone Repository in Container Volume...
```

    3.2. Enter the URL of your new repo, e.g. https://github.com/olilarkin/iPlug2OOS or sign in with github and select the URL from the list.

    3.3. Select **“create a new volume”**

    3.4. Wait a little while while the container is built …

    3.5. Open the **iPlug2OOS.code-workspace**



This folder contains a workspace file 'iPlug2OOS.code-workspace'. Do you want to open it? Learn more about workspace files.

4. Set up the repository

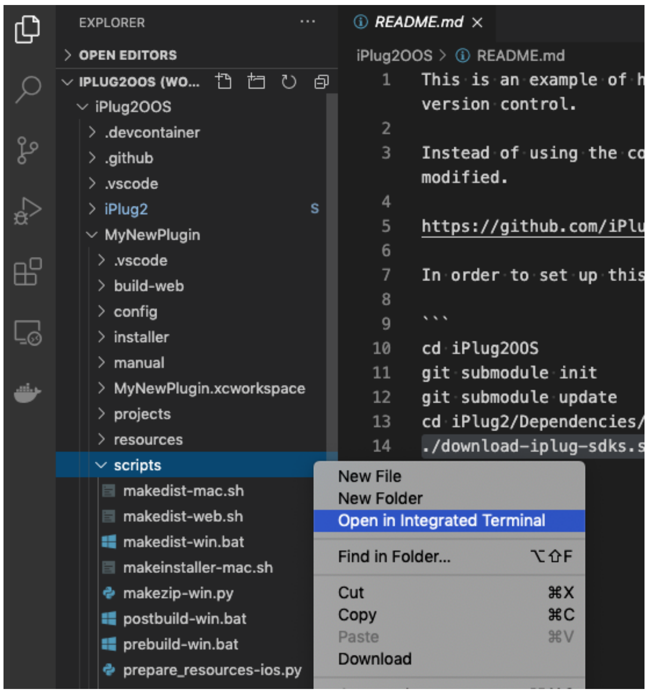
Now we have a container set up in VSCode, which contains a git submodule for the iPlug2 source code, alongside a template folder for our project “TemplateProject”. The container initialization script checks out the iPlug2 submodule and downloads the iPlug2 SDK dependencies (e.g. the WAM SDK) into iPlug2/Dependencies. Now we want to rename the template project:

    4.6. Open a new terminal. Right click the top level “iPlug2OOS” folder in the workspace explorer and select **“Open in integrated terminal ...”.** There is a python script at the top level of the iPlug2OOS repo called **duplicate.py** which you can use to rename the template project “TemplateProject” with your own PROJECT_NAME and also specify your MANUFACTURER_NAME.

Now you should see code in the “iPlug2” subfolder and a new folder PROJECT_NAME in place of TemplateProject.

5. Build the WAM

5.1. Navigate to the folder **scripts** inside the **TemplateProject** subfolder and launch a terminal there.



```
This is an example of h
version control.

Instead of using the co
modified.

https://github.com/iPlu

In order to set up this

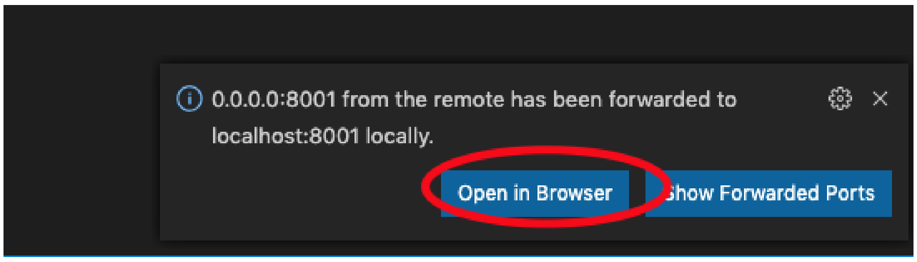
```

cd iPlug2OOS
git submodule init
git submodule update
cd iPlug2/Dependencies/
./download-iplug-sdks.s
```

5.2. Now you can execute the **makedist-web.sh** script with the argument **container.** The argument is required because we are running the script inside a headless container machine where we don’t have a web browser.

```
./makedist-web.sh container
```

5.3. When it is built, it runs a web server. Click “Open in Browser”, to open the forwarded ports in your web browser.


0.0.0.0:8001 from the remote has been forwarded to localhost:8001 locally.

Congratulations! You should now have built a WAM. Since the command in 5.2 is the default Build Task in the VSCode workspace, you can also launch it from the command palette or via the keyboard shortcut CMD/CTRL-SHIFT-B.

6. Edit the code

6.1. Kill the web server by pressing **CTRL-C** in the terminal.
6.2. Locate the file PROJECT_NAME/**PROJECT_NAME.cpp,** and edit the background color, and save the file…
```

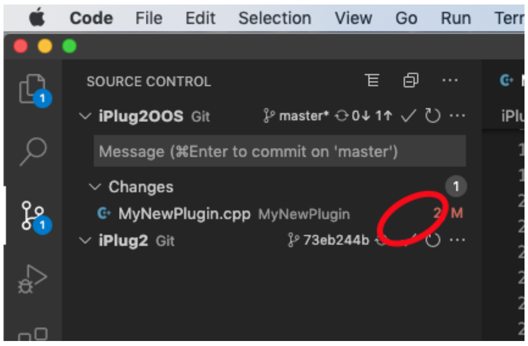
```
pGraphics->AttachControl(new IVKnobControl(control
```

6.3. Now you can repeat step 5.2, to rebuild the WAM with **makedist-web.sh** . If you made any mistakes in your C++ code, the emscripten compiler will tell you!

7. Commit changes and publish in the cloud

When you are happy with the changes, we need to commit them to version control, and push to github in order to trigger a build.

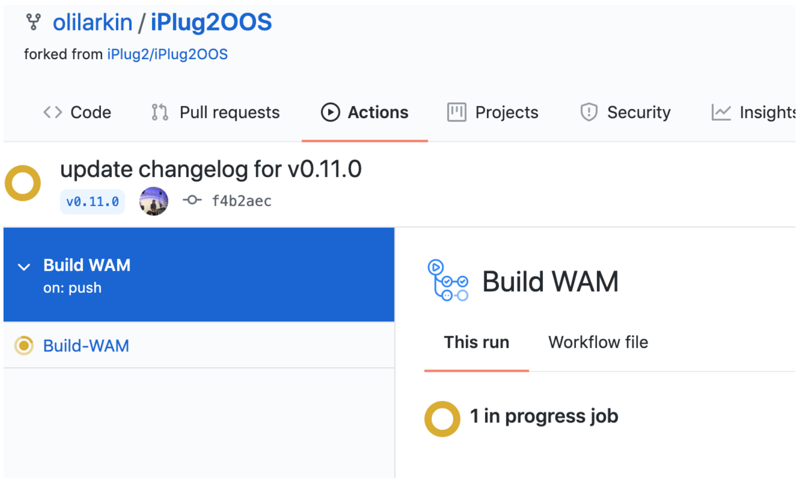
7.1. You can commit them either using the command line terminal, or the VSCode Source Control pane.

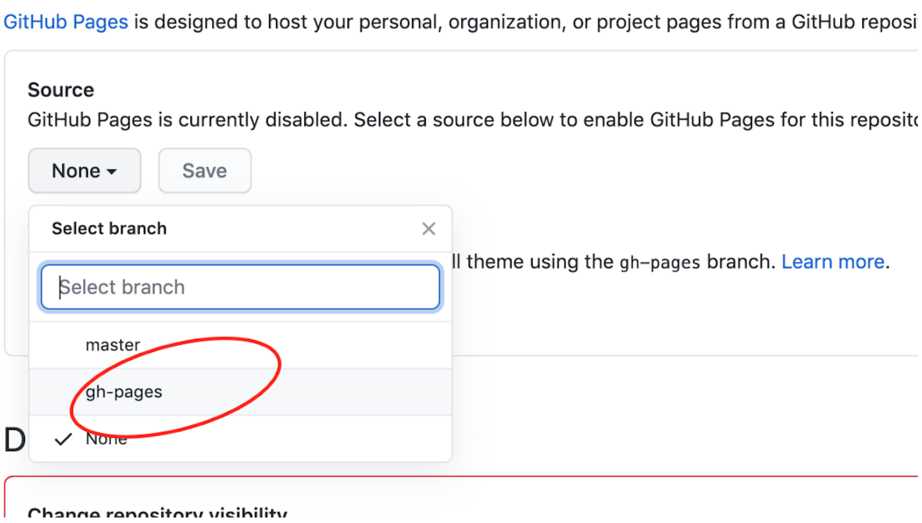

Code File Edit Selection View Go Run Terr
SOURCE CONTROL
iPlug2OOS Git master* 0↓ 1↑
Message (⌘Enter to commit on 'master')
Changes 1
MyNewPlugin.cpp MyNewPlugin 2 M
iPlug2 Git 73eb244b

7.2. There is a script in the iPlug2OOS root folder **bump_version.py** which you can use to tag a new version of your project, and push to github in order to trigger a build. Navigate to the iPlug2OOS root folder and type…

```
./bump_version.py minor
```

Follow the instructions. You'll need to use VIM when prompted for a commit message, and if you want to edit the changelog. Vim is a command line text editor which is notoriously confusing! All you need to know is that you start in **command mode.** Press the key **i** to enter **insert mode**. Now you can edit the text file. When you are happy, press the escape key to exit insert mode. Now type **:wq**, which is the command for "write and quit". After quitting vim the script completes its tasks. When everything has been pushed to the cloud it should trigger a build on your github project, which you can see by visiting the Actions page:




GitHub Pages is designed to host your personal, organization, or project pages from a GitHub reposi

Source

GitHub Pages is currently disabled. Select a source below to enable GitHub Pages for this reposito

None
Save
Select branch
Select branch
master
gh-pages
✓ None
ll theme using the gh-pages branch. Learn more.
D
Change repository visibility

7.4. Wait a little while for github pages to be updated. Now your new project should be hosted at

https://YOUR_GITHUB_USERNAME.github.io/PROJECT_NAME/

---

[1] continuous integration/deployment