Anima V3 Deployment Manual

Deployed Client System requirements:

- Windows 8 and above
- Installed Microsoft Visual C++ Redistributable
- Installed .NET Framework 4.7.2 or above
- Required Internet access for the first time running

There is already a release version of software for users to download. The software only can be run when the system requirements above are satisfied.

Double click on the ‘AnimaV3.exe’ in the zip file downloaded, and Anima V3 will be ready to go!

Note: The first time start-up might take about two minutes, please be patient and do not close the application. If Windows protection pops up to say virus warning, just run anyway.

If any functionality is required to add and upgrade, here are steps below to deploy a new build.

Project Architecture:

- Backend
- Frontend
- FFmpeg(Optional)

In the project repo, it already includes the backend and frontend parts. Basically, deployment is to compile them to generate a build individually, and then integrate them to a specific directory.

Steps:

Backend:

Go into the ‘/backend’ directory.

Open the ‘pyoxidizer.bzl’, this is the configuration file to build backend.

In ‘make_install’ this section, modify the path of templates. Replace the original path with the actual absolute path of ‘/backend/ANIMA’ in your system.

```
284
285 def make_install(exe):
286     # Create an object that represents our installed application file layout.
287     files = FileManifest()
288
289     # Add the generated executable to our install layout in the root directory.
290     files.add_python_resource(".", exe)
291     templates = glob(["E:\\anima\\wholeAnima\\backend\\ANIMA\\**\\*"], strip_prefix="E:\\anima\\wholeAnima\\backend\\ANIMA")
292     files.add_manifest(templates)
293
294     return files
```

For example, assume the ‘ANIMA’ path in your system is ‘E:/anima/backend/ANIMA’, you should modify code like ‘templates = glob([“E:\\anima\\backend\\ANIMA***”], strip_prefix= “E:\\anima\\backend\\ANIMA”)’.

The difference of two paths in templates is that the first path has ‘***’ at the end, which is necessary, but the second path does not. When you modify path, please use double right slash ‘\\’ as the path character.

After you done the steps above, open the terminal and go into the ‘/backend’ directory. Execute ‘**pyoxidizer build install’** this command, then Pyoxidizer would start to build the backend.

Once the build is generated, a directory of ‘/backend/ANIMA/build/x86_64-pc-windows-msvc/debug/install’ should appear, which is the build of backend.

Before integrating the backend, it is suggested to clean the backend source code and unnecessary files to make project folder clear. But even if you do not clean source code, the backend build still does work. For source code cleaning, delete all source code files and unnecessary files such as .py and .bzl in ‘/backend/ANIMA’(exclude the folder itself), and only keep JSON files.

Copy all files in ‘/backend/ANIMA/build/x86_64-pc-windows-msvc/debug/install’ into ‘/backend/ANIMA/.

NOTICE: Until March 2024, Pyoxidizer had a bug on handling ‘MeCab’. The build from Pyoxidizer has a missed DLL file, thus we must copy it manually.

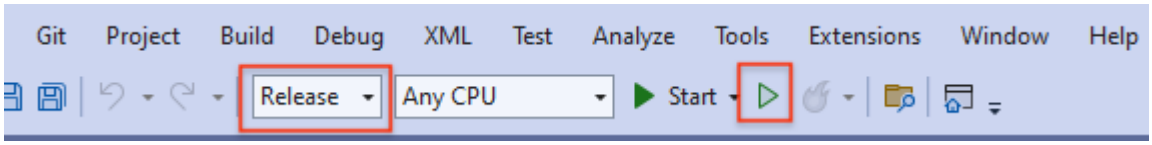
Go into the ‘MeCab’ directory of PyPI libraries in your system. This path is usually ‘Lib/site-packages/MeCab’ under the Python Interpreter directory. If you are using Conda or Venv environment, this directory should be found in the environment directory.

Copy ‘Lib/site-packages/MeCab/libmecab.dll’ into ‘/backend/ANIMA’, then the backend build task is finished.

Frontend:

The frontend build needs **Microsoft Visual Studio** to compile, so make sure Visual Studio has been installed before building the frontend. When install the Visual Studio, make sure required components for .NET Framework 4.7.2 WPF application are installed.

Use Visual Studio to open ‘dotnetAnima.sln’.



In the top toolbar, change the build mode from ‘Debug’ to ‘Release, then click the green hollow triangle button once. If no error, the software should be running.

After the first build, Release files should appear in the root directory, which is the frontend build.

Before integrating the frontend, it is suggested to clean the frontend source code and unnecessary files to make project folder clear. But even if you do not clean source code, the frontend build still does work. For source code cleaning, delete all source code files such as xaml and cs files, but leave all JSON files, txt files, and static resources files such as icons and images.

Open ‘**dotnetAnima.exe**’ to run the software, then the frontend build task is finished.

To improve the user experience, a shortcut can be created to quickly open the software.

FFmpeg(Optional):

Sometimes other media encoders might be installed on the system, even if the FFmpeg is not installed, the software can still work. But it is still recommended to install the FFmpeg for the compatibility.

Download the FFmpeg build from the official website, but notice that the FFmpeg version should be 4.X. In that build, it should have 4 directories, ‘/include’, ‘/bin’, ‘/doc’, and ‘/lib’. Once downloaded, go into the ‘bin’ directory, and check if ‘avdevice-58.dll’ exists or not. If

ffmpeg-n4.4-latest-linux64-gpl-4.4.tar.xz	82.2 MB	10 hours ago
ffmpeg-n4.4-latest-linux64-gpl-shared-4.4.tar.xz	33.2 MB	10 hours ago
ffmpeg-n4.4-latest-linux64-lgpl-4.4.tar.xz	70.2 MB	10 hours ago
ffmpeg-n4.4-latest-linux64-lgpl-shared-4.4.tar.xz	28.1 MB	10 hours ago
ffmpeg-n4.4-latest-linuxarm64-gpl-4.4.tar.xz	65.9 MB	10 hours ago
ffmpeg-n4.4-latest-linuxarm64-gpl-shared-4.4.tar.xz	27.5 MB	10 hours ago
ffmpeg-n4.4-latest-linuxarm64-lgpl-4.4.tar.xz	58.6 MB	10 hours ago
ffmpeg-n4.4-latest-linuxarm64-lgpl-shared-4.4.tar.xz	24 MB	10 hours ago
ffmpeg-n4.4-latest-win64-gpl-4.4.zip	109 MB	10 hours ago
ffmpeg-n4.4-latest-win64-gpl-shared-4.4.zip	42.4 MB	10 hours ago

‘avdevice-58.dll’ is not found, or its suffix number is not 58, or there are no ‘include’ and ‘lib’ directories. It means that you downloaded an incorrect version. The correct version should have ‘avdevice-58.dll’ under the ‘bin’ directory.

Copy all files in ‘/bin’ into ‘/backend/ANIMA’ directory(not include ‘bin’ folder itself) in the backend, and then copy ‘/include’, ‘/doc’ and ‘/lib’ these 3 folders into ‘/backend/ANIMA’ directory.

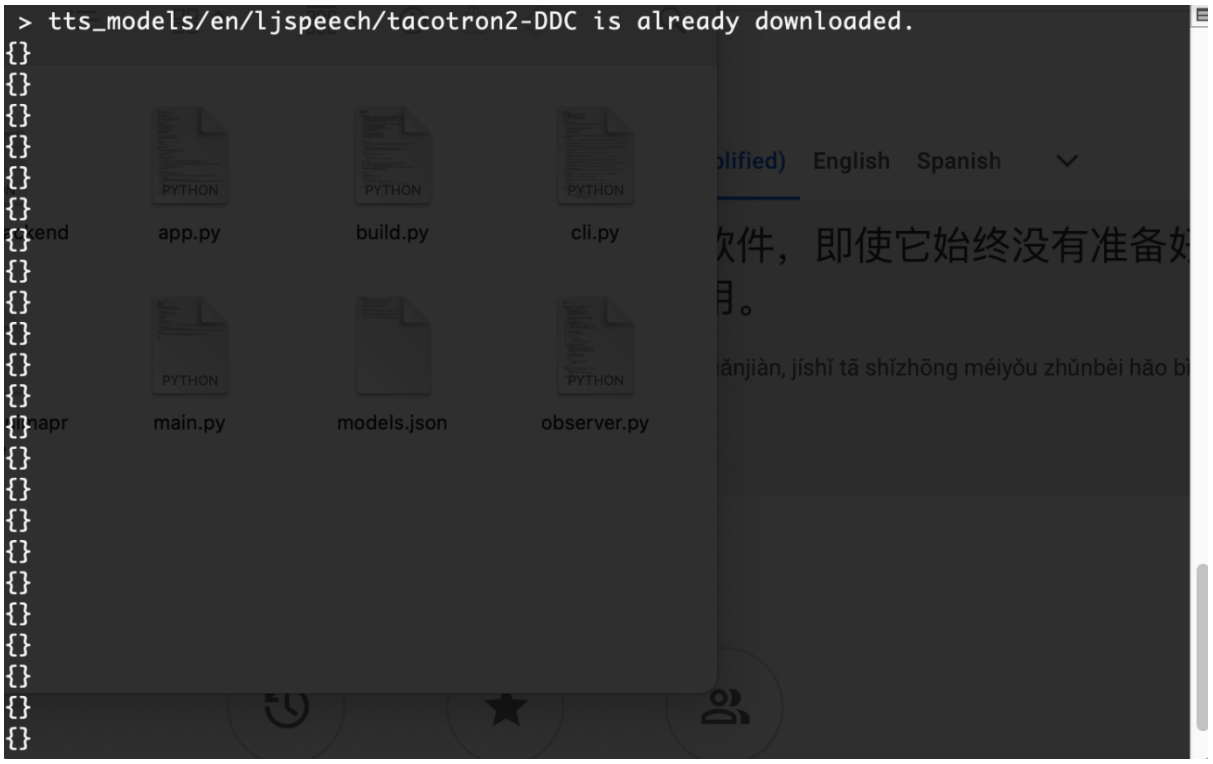
Debug Mode:

When the software is opened, usually a few seconds are needed to start up all processes(the first time of running would spend longer time).

But in any case, the startup time should not exceed two minutes. If the software is not yet ready to start within two minutes, there must be something wrong. Usually, a warning window would pop up if any problem results in failed startup. However, if no pop-up warning window, and software is always not ready, we can use debug mode to check.

How to entry debug mode:

Firstly, open the software, even if it is always not ready and ‘voicebank’ button is disabled. Keep running the software, and go into the ‘/backend/ANIMA/’ directory. Open ‘anima.exe’, then a terminal window should be opened like in this picture.



The normal debug message should look like this above. If not, please follow the error message to fix.

Troubleshooting:

‘avdevice-58.dll’ cannot be found:

FFmpeg needs to be installed, please follow the instructions of FFmpeg to install.

A page is frozen for a long time:

If the home page is frozen and the button is not ready to click for a long time, firstly try to close anti-virus software and run as administrator.

If the step above does not work or other pages frozen, and no warning window pop up, enter the Debug Mode to check debug message.

In the debug window, an error message of “Model not found”:

Go into the ‘/User/YourName/.AppData/Local’ directory, delete ‘tts’ this folder and run the software again(‘YourName’ is your username).

In the debug window, an error message of “ffmpeg not found”:

FFmpeg is not installed correctly. If the software works normally, then this message can be ignored. Otherwise, follow the steps above to correctly install the FFmpeg.