#include "utils.h"

#include <flutter\_windows.h>

#include <io.h>

#include <stdio.h>

#include <windows.h>

#include <iostream>

void CreateAndAttachConsole() {

if (::AllocConsole()) {

FILE \*unused;

if (freopen\_s(&unused, "CONOUT$", "w", stdout)) {

\_dup2(\_fileno(stdout), 1);

}

if (freopen\_s(&unused, "CONOUT$", "w", stderr)) {

\_dup2(\_fileno(stdout), 2);

}

std::ios::sync\_with\_stdio();

FlutterDesktopResyncOutputStreams();

}

}

std::vector<std::string> GetCommandLineArguments() {

// Convert the UTF-16 command line arguments to UTF-8 for the Engine to use.

int argc;

wchar\_t\*\* argv = ::CommandLineToArgvW(::GetCommandLineW(), &argc);

if (argv == nullptr) {

return std::vector<std::string>();

}

std::vector<std::string> command\_line\_arguments;

// Skip the first argument as it's the binary name.

for (int i = 1; i < argc; i++) {

command\_line\_arguments.push\_back(Utf8FromUtf16(argv[i]));

}

::LocalFree(argv);

return command\_line\_arguments;

}

std::string Utf8FromUtf16(const wchar\_t\* utf16\_string) {

if (utf16\_string == nullptr) {

return std::string();

}

int target\_length = ::WideCharToMultiByte(

CP\_UTF8, WC\_ERR\_INVALID\_CHARS, utf16\_string,

-1, nullptr, 0, nullptr, nullptr)

-1; // remove the trailing null character

int input\_length = (int)wcslen(utf16\_string);

std::string utf8\_string;

if (target\_length <= 0 || target\_length > utf8\_string.max\_size()) {

return utf8\_string;

}

utf8\_string.resize(target\_length);

int converted\_length = ::WideCharToMultiByte(

CP\_UTF8, WC\_ERR\_INVALID\_CHARS, utf16\_string,

input\_length, utf8\_string.data(), target\_length, nullptr, nullptr);

if (converted\_length == 0) {

return std::string();

}

return utf8\_string;

}