// Copyright 2013 The Flutter Authors. All rights reserved.

// Use of this source code is governed by a BSD-style license that can be

// found in the LICENSE file.

#include "include/flutter/plugin\_registrar.h"

#include <iostream>

#include <map>

#include "binary\_messenger\_impl.h"

#include "include/flutter/engine\_method\_result.h"

#include "include/flutter/method\_channel.h"

#include "texture\_registrar\_impl.h"

namespace flutter {

// ===== PluginRegistrar =====

PluginRegistrar::PluginRegistrar(FlutterDesktopPluginRegistrarRef registrar)

: registrar\_(registrar) {

auto core\_messenger = FlutterDesktopPluginRegistrarGetMessenger(registrar\_);

messenger\_ = std::make\_unique<BinaryMessengerImpl>(core\_messenger);

auto texture\_registrar =

FlutterDesktopRegistrarGetTextureRegistrar(registrar\_);

texture\_registrar\_ =

std::make\_unique<TextureRegistrarImpl>(texture\_registrar);

}

PluginRegistrar::~PluginRegistrar() {

// This must always be the first call.

ClearPlugins();

// Explicitly cleared to facilitate testing of destruction order.

messenger\_.reset();

}

void PluginRegistrar::AddPlugin(std::unique\_ptr<Plugin> plugin) {

plugins\_.insert(std::move(plugin));

}

void PluginRegistrar::ClearPlugins() {

plugins\_.clear();

}

// ===== PluginRegistrarManager =====

// static

PluginRegistrarManager\* PluginRegistrarManager::GetInstance() {

static PluginRegistrarManager\* instance = new PluginRegistrarManager();

return instance;

}

PluginRegistrarManager::PluginRegistrarManager() = default;

// static

void PluginRegistrarManager::OnRegistrarDestroyed(

FlutterDesktopPluginRegistrarRef registrar) {

GetInstance()->registrars()->erase(registrar);

}

} // namespace flutter