// 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.

#ifndef FLUTTER\_SHELL\_PLATFORM\_COMMON\_PUBLIC\_FLUTTER\_MESSENGER\_H\_

#define FLUTTER\_SHELL\_PLATFORM\_COMMON\_PUBLIC\_FLUTTER\_MESSENGER\_H\_

#include <stdbool.h>

#include <stddef.h>

#include <stdint.h>

#include "flutter\_export.h"

#if defined(\_\_cplusplus)

extern "C" {

#endif // defined(\_\_cplusplus)

// Opaque reference to a Flutter engine messenger.

typedef struct FlutterDesktopMessenger\* FlutterDesktopMessengerRef;

// Opaque handle for tracking responses to messages.

typedef struct \_FlutterPlatformMessageResponseHandle

FlutterDesktopMessageResponseHandle;

// The callback expected as a response of a binary message.

typedef void (\*FlutterDesktopBinaryReply)(const uint8\_t\* data,

size\_t data\_size,

void\* user\_data);

// A message received from Flutter.

typedef struct {

// Size of this struct as created by Flutter.

size\_t struct\_size;

// The name of the channel used for this message.

const char\* channel;

// The raw message data.

const uint8\_t\* message;

// The length of |message|.

size\_t message\_size;

// The response handle. If non-null, the receiver of this message must call

// FlutterDesktopSendMessageResponse exactly once with this handle.

const FlutterDesktopMessageResponseHandle\* response\_handle;

} FlutterDesktopMessage;

// Function pointer type for message handler callback registration.

//

// The user data will be whatever was passed to FlutterDesktopSetMessageHandler

// for the channel the message is received on.

typedef void (\*FlutterDesktopMessageCallback)(

FlutterDesktopMessengerRef /\* messenger \*/,

const FlutterDesktopMessage\* /\* message\*/,

void\* /\* user data \*/);

// Sends a binary message to the Flutter side on the specified channel.

FLUTTER\_EXPORT bool FlutterDesktopMessengerSend(

FlutterDesktopMessengerRef messenger,

const char\* channel,

const uint8\_t\* message,

const size\_t message\_size);

// Sends a binary message to the Flutter side on the specified channel.

// The |reply| callback will be executed when a response is received.

FLUTTER\_EXPORT bool FlutterDesktopMessengerSendWithReply(

FlutterDesktopMessengerRef messenger,

const char\* channel,

const uint8\_t\* message,

const size\_t message\_size,

const FlutterDesktopBinaryReply reply,

void\* user\_data);

// Sends a reply to a FlutterDesktopMessage for the given response handle.

//

// Once this has been called, |handle| is invalid and must not be used again.

FLUTTER\_EXPORT void FlutterDesktopMessengerSendResponse(

FlutterDesktopMessengerRef messenger,

const FlutterDesktopMessageResponseHandle\* handle,

const uint8\_t\* data,

size\_t data\_length);

// Registers a callback function for incoming binary messages from the Flutter

// side on the specified channel.

//

// Replaces any existing callback. Provide a null handler to unregister the

// existing callback.

//

// If |user\_data| is provided, it will be passed in |callback| calls.

FLUTTER\_EXPORT void FlutterDesktopMessengerSetCallback(

FlutterDesktopMessengerRef messenger,

const char\* channel,

FlutterDesktopMessageCallback callback,

void\* user\_data);

// Increments the reference count for the |messenger|.

//

// Operation is thread-safe.

//

// See also: |FlutterDesktopMessengerRelease|

FLUTTER\_EXPORT FlutterDesktopMessengerRef FlutterDesktopMessengerAddRef(

FlutterDesktopMessengerRef messenger);

// Decrements the reference count for the |messenger|.

//

// Operation is thread-safe.

//

// See also: |FlutterDesktopMessengerAddRef|

FLUTTER\_EXPORT void FlutterDesktopMessengerRelease(

FlutterDesktopMessengerRef messenger);

// Returns `true` if the |FlutterDesktopMessengerRef| still references a running

// engine.

//

// This check should be made inside of a |FlutterDesktopMessengerLock| and

// before any other calls are made to the FlutterDesktopMessengerRef when using

// it from a thread other than the platform thread.

FLUTTER\_EXPORT bool FlutterDesktopMessengerIsAvailable(

FlutterDesktopMessengerRef messenger);

// Locks the `FlutterDesktopMessengerRef` ensuring that

// |FlutterDesktopMessengerIsAvailable| does not change while locked.

//

// All calls to the FlutterDesktopMessengerRef from threads other than the

// platform thread should happen inside of a lock.

//

// Operation is thread-safe.

//

// Returns the |messenger| value.

//

// See also: |FlutterDesktopMessengerUnlock|

FLUTTER\_EXPORT FlutterDesktopMessengerRef FlutterDesktopMessengerLock(

FlutterDesktopMessengerRef messenger);

// Unlocks the `FlutterDesktopMessengerRef`.

//

// Operation is thread-safe.

//

// See also: |FlutterDesktopMessengerLock|

FLUTTER\_EXPORT void FlutterDesktopMessengerUnlock(

FlutterDesktopMessengerRef messenger);

#if defined(\_\_cplusplus)

} // extern "C"

#endif

#endif // FLUTTER\_SHELL\_PLATFORM\_COMMON\_PUBLIC\_FLUTTER\_MESSENGER\_H\_