// 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\_CLIENT\_WRAPPER\_INCLUDE\_FLUTTER\_EVENT\_STREAM\_HANDLER\_H\_

#define FLUTTER\_SHELL\_PLATFORM\_COMMON\_CLIENT\_WRAPPER\_INCLUDE\_FLUTTER\_EVENT\_STREAM\_HANDLER\_H\_

#include <memory>

#include <string>

#include "event\_sink.h"

namespace flutter {

class EncodableValue;

template <typename T = EncodableValue>

struct StreamHandlerError {

const std::string error\_code;

const std::string error\_message;

const std::unique\_ptr<T> error\_details;

StreamHandlerError(const std::string& error\_code,

const std::string& error\_message,

std::unique\_ptr<T>&& error\_details)

: error\_code(error\_code),

error\_message(error\_message),

error\_details(std::move(error\_details)) {}

};

// Handler for stream setup and teardown requests.

// Implementations must be prepared to accept sequences of alternating calls to

// OnListen() and OnCancel(). Implementations should ideally consume no

// resources when the last such call is not OnListen(). In typical situations,

// this means that the implementation should register itself with

// platform-specific event sources OnListen() and deregister again OnCancel().

template <typename T = EncodableValue>

class StreamHandler {

public:

StreamHandler() = default;

virtual ~StreamHandler() = default;

// Prevent copying.

StreamHandler(StreamHandler const&) = delete;

StreamHandler& operator=(StreamHandler const&) = delete;

// Handles a request to set up an event stream. Returns nullptr on success,

// or an error on failure.

// |arguments| is stream configuration arguments and

// |events| is an EventSink for emitting events to the Flutter receiver.

std::unique\_ptr<StreamHandlerError<T>> OnListen(

const T\* arguments,

std::unique\_ptr<EventSink<T>>&& events) {

return OnListenInternal(arguments, std::move(events));

}

// Handles a request to tear down the most recently created event stream.

// Returns nullptr on success, or an error on failure.

// |arguments| is stream configuration arguments.

std::unique\_ptr<StreamHandlerError<T>> OnCancel(const T\* arguments) {

return OnCancelInternal(arguments);

}

protected:

// Implementation of the public interface, to be provided by subclasses.

virtual std::unique\_ptr<StreamHandlerError<T>> OnListenInternal(

const T\* arguments,

std::unique\_ptr<EventSink<T>>&& events) = 0;

// Implementation of the public interface, to be provided by subclasses.

virtual std::unique\_ptr<StreamHandlerError<T>> OnCancelInternal(

const T\* arguments) = 0;

};

} // namespace flutter

#endif // FLUTTER\_SHELL\_PLATFORM\_COMMON\_CLIENT\_WRAPPER\_INCLUDE\_FLUTTER\_EVENT\_STREAM\_HANDLER\_H\_