// 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\_STANDARD\_MESSAGE\_CODEC\_H\_

#define FLUTTER\_SHELL\_PLATFORM\_COMMON\_CLIENT\_WRAPPER\_INCLUDE\_FLUTTER\_STANDARD\_MESSAGE\_CODEC\_H\_

#include <memory>

#include "encodable\_value.h"

#include "message\_codec.h"

#include "standard\_codec\_serializer.h"

namespace flutter {

// A binary message encoding/decoding mechanism for communications to/from the

// Flutter engine via message channels.

class StandardMessageCodec : public MessageCodec<EncodableValue> {

public:

// Returns an instance of the codec, optionally using a custom serializer to

// add support for more types.

//

// If provided, |serializer| must be long-lived. If no serializer is provided,

// the default will be used.

//

// The instance returned for a given |serializer| will be shared, and

// any instance returned from this will be long-lived, and can be safely

// passed to, e.g., channel constructors.

static const StandardMessageCodec& GetInstance(

const StandardCodecSerializer\* serializer = nullptr);

~StandardMessageCodec();

// Prevent copying.

StandardMessageCodec(StandardMessageCodec const&) = delete;

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

protected:

// |flutter::MessageCodec|

std::unique\_ptr<EncodableValue> DecodeMessageInternal(

const uint8\_t\* binary\_message,

const size\_t message\_size) const override;

// |flutter::MessageCodec|

std::unique\_ptr<std::vector<uint8\_t>> EncodeMessageInternal(

const EncodableValue& message) const override;

private:

// Instances should be obtained via GetInstance.

explicit StandardMessageCodec(const StandardCodecSerializer\* serializer);

const StandardCodecSerializer\* serializer\_;

};

} // namespace flutter

#endif // FLUTTER\_SHELL\_PLATFORM\_COMMON\_CLIENT\_WRAPPER\_INCLUDE\_FLUTTER\_STANDARD\_MESSAGE\_CODEC\_H\_