// 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\_TEXTURE\_REGISTRAR\_H\_

#define FLUTTER\_SHELL\_PLATFORM\_COMMON\_CLIENT\_WRAPPER\_INCLUDE\_FLUTTER\_TEXTURE\_REGISTRAR\_H\_

#include <flutter\_texture\_registrar.h>

#include <cstdint>

#include <functional>

#include <memory>

#include <utility>

#include <variant>

namespace flutter {

// A pixel buffer texture.

class PixelBufferTexture {

public:

// A callback used for retrieving pixel buffers.

typedef std::function<const FlutterDesktopPixelBuffer\*(size\_t width,

size\_t height)>

CopyBufferCallback;

// Creates a pixel buffer texture that uses the provided |copy\_buffer\_cb| to

// retrieve the buffer.

// As the callback is usually invoked from the render thread, the callee must

// take care of proper synchronization. It also needs to be ensured that the

// returned buffer isn't released prior to unregistering this texture.

explicit PixelBufferTexture(CopyBufferCallback copy\_buffer\_callback)

: copy\_buffer\_callback\_(std::move(copy\_buffer\_callback)) {}

// Returns the callback-provided FlutterDesktopPixelBuffer that contains the

// actual pixel data. The intended surface size is specified by |width| and

// |height|.

const FlutterDesktopPixelBuffer\* CopyPixelBuffer(size\_t width,

size\_t height) const {

return copy\_buffer\_callback\_(width, height);

}

private:

const CopyBufferCallback copy\_buffer\_callback\_;

};

// A GPU surface-based texture.

class GpuSurfaceTexture {

public:

// A callback used for retrieving surface descriptors.

typedef std::function<

const FlutterDesktopGpuSurfaceDescriptor\*(size\_t width, size\_t height)>

ObtainDescriptorCallback;

GpuSurfaceTexture(FlutterDesktopGpuSurfaceType surface\_type,

ObtainDescriptorCallback obtain\_descriptor\_callback)

: surface\_type\_(surface\_type),

obtain\_descriptor\_callback\_(std::move(obtain\_descriptor\_callback)) {}

// Returns the callback-provided FlutterDesktopGpuSurfaceDescriptor that

// contains the surface handle. The intended surface size is specified by

// |width| and |height|.

const FlutterDesktopGpuSurfaceDescriptor\* ObtainDescriptor(

size\_t width,

size\_t height) const {

return obtain\_descriptor\_callback\_(width, height);

}

// Gets the surface type.

FlutterDesktopGpuSurfaceType surface\_type() const { return surface\_type\_; }

private:

const FlutterDesktopGpuSurfaceType surface\_type\_;

const ObtainDescriptorCallback obtain\_descriptor\_callback\_;

};

// The available texture variants.

// Only PixelBufferTexture is currently implemented.

// Other variants are expected to be added in the future.

typedef std::variant<PixelBufferTexture, GpuSurfaceTexture> TextureVariant;

// An object keeping track of external textures.

//

// Thread safety:

// It's safe to call the member methods from any thread.

class TextureRegistrar {

public:

virtual ~TextureRegistrar() = default;

// Registers a |texture| object and returns the ID for that texture.

virtual int64\_t RegisterTexture(TextureVariant\* texture) = 0;

// Notifies the flutter engine that the texture object corresponding

// to |texure\_id| needs to render a new frame.

//

// For PixelBufferTextures, this will effectively make the engine invoke

// the callback that was provided upon creating the texture.

virtual bool MarkTextureFrameAvailable(int64\_t texture\_id) = 0;

// Asynchronously unregisters an existing texture object.

// Upon completion, the optional |callback| gets invoked.

virtual void UnregisterTexture(int64\_t texture\_id,

std::function<void()> callback) = 0;

// Unregisters an existing texture object.

// DEPRECATED: Use UnregisterTexture(texture\_id, optional\_callback) instead.

virtual bool UnregisterTexture(int64\_t texture\_id) = 0;

};

} // namespace flutter

#endif // FLUTTER\_SHELL\_PLATFORM\_COMMON\_CLIENT\_WRAPPER\_INCLUDE\_FLUTTER\_TEXTURE\_REGISTRAR\_H\_