Writing multi window applications

Ricardo De Peña - github.com/rdepena
ricardo@openfin.co

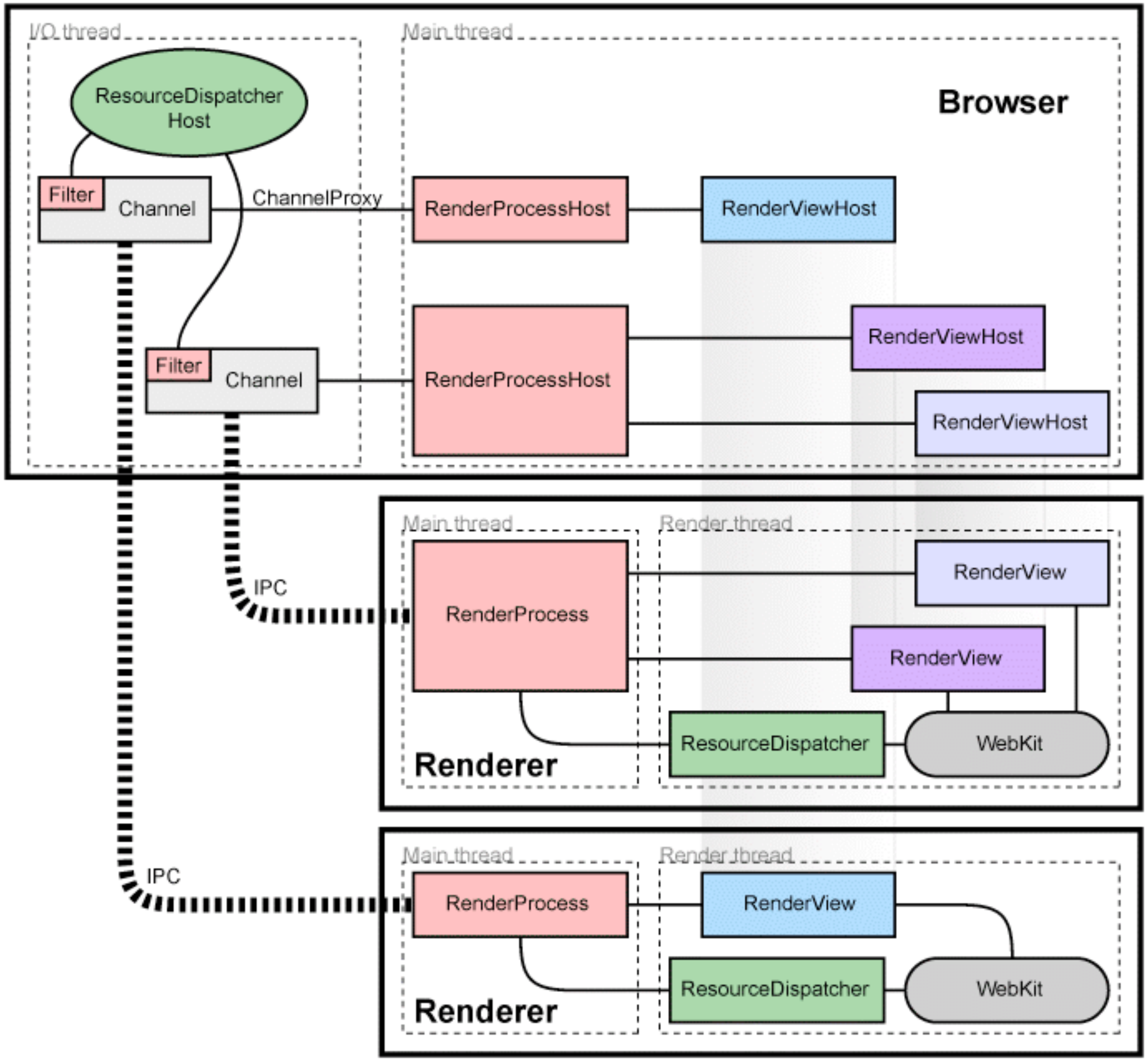
Challenges

- Not the traditional model for Web
- Threading and process architecture take the forefront
- Moving data efficiently between windows is key for performance

Good News

- Standard Web API’s are flexible enough

“If you wish to make a Web Application from scratch you must first invent the renderer”


I/O thread
Main thread
Browser
ResourceDispatcher Host
Filter
Channel
ChannelProxy
RenderProcessHost
RenderViewHost
Filter
Channel
RenderProcessHost
RenderViewHost
RenderViewHost
Main thread
Render thread
IPC
RenderProcess
RenderView
RenderView
ResourceDispatcher
WebKit
Renderer
Main thread
Render thread
IPC
RenderProcess
RenderView
ResourceDispatcher
WebKit
Renderer

Render Process

- In OpenFin are called Application
- Execution context (main thread, memory)
- Collection of windows

Demo Time

Two ways to spawn a Window

- Child Window (OpenFin Window)
- New Render Process (OpenFin Application)

Child Window

- Can be created with either window.open or OpenFin new Window
- Shared main thread
- Shared memory
- Can easily share DOM
- Can easily share data

Demo time

Web Workers

- Execution done on a background thread
- Isolated Javascript context
- Transferable Interface allows to share data without creating copies
- ArrayBuffers, MessagePorts and ImageBitmap types implement the Transferable interface

Demo time

New Render Process

- Can be created with the OpenFin new Application
- Isolated Javascript context
- Cannot simply share data with them
- Larger memory footprint

Demo time

Shared Workers

- Execution done on a background thread
- Isolated Javascript context
- Can share data between Render processes as long as they share the same origin

Demo time

Patterns Emerge

- Start with a single Renderer with multiple windows
- Avoid Late 90s dialog boxes
- Use Web Workers when you need thread isolation on cpu bound tasks
- Create Applications when you need process level isolation
- Use Shared Workers instead of Web Workers in a multi Renderer environment
- Use Service Workers to proxy data to multiple renderers and add the ability to use the application offline
- Extra processing is always available with Native Adapters and the InterApplication Bus

Questions?

https://github.com/openfin/meetup-process-model