Embrace Modern Technology:

Using HTML 5 for GUI in C++

by **Borislav Stanimirov** / **@stanimirovb**

Hello, World

```
#include <iostream>

int main()
{
    std::cout << "Hi, I'm Borislav!\n";
    std::cout << "These slides are here: https://is.gd/html5gui\n";
    return 0;
}
```

Borislav Stanimirov

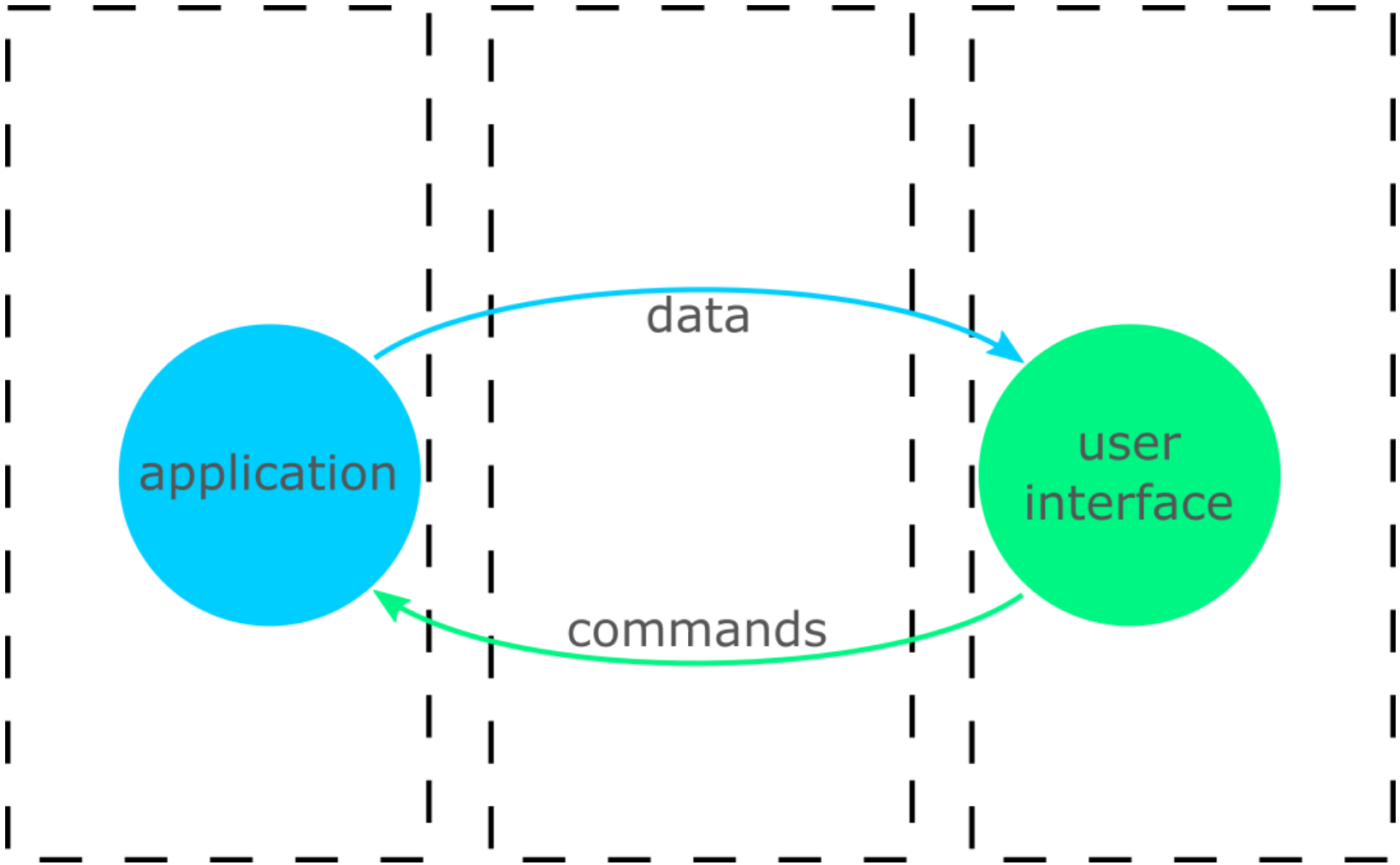
- Mostly a **C++** programmer
- Mostly a **game** programmer
- Recently a **medical software** programmer
- **Open-source** programmer
- **github.com/iboB**

About this talk

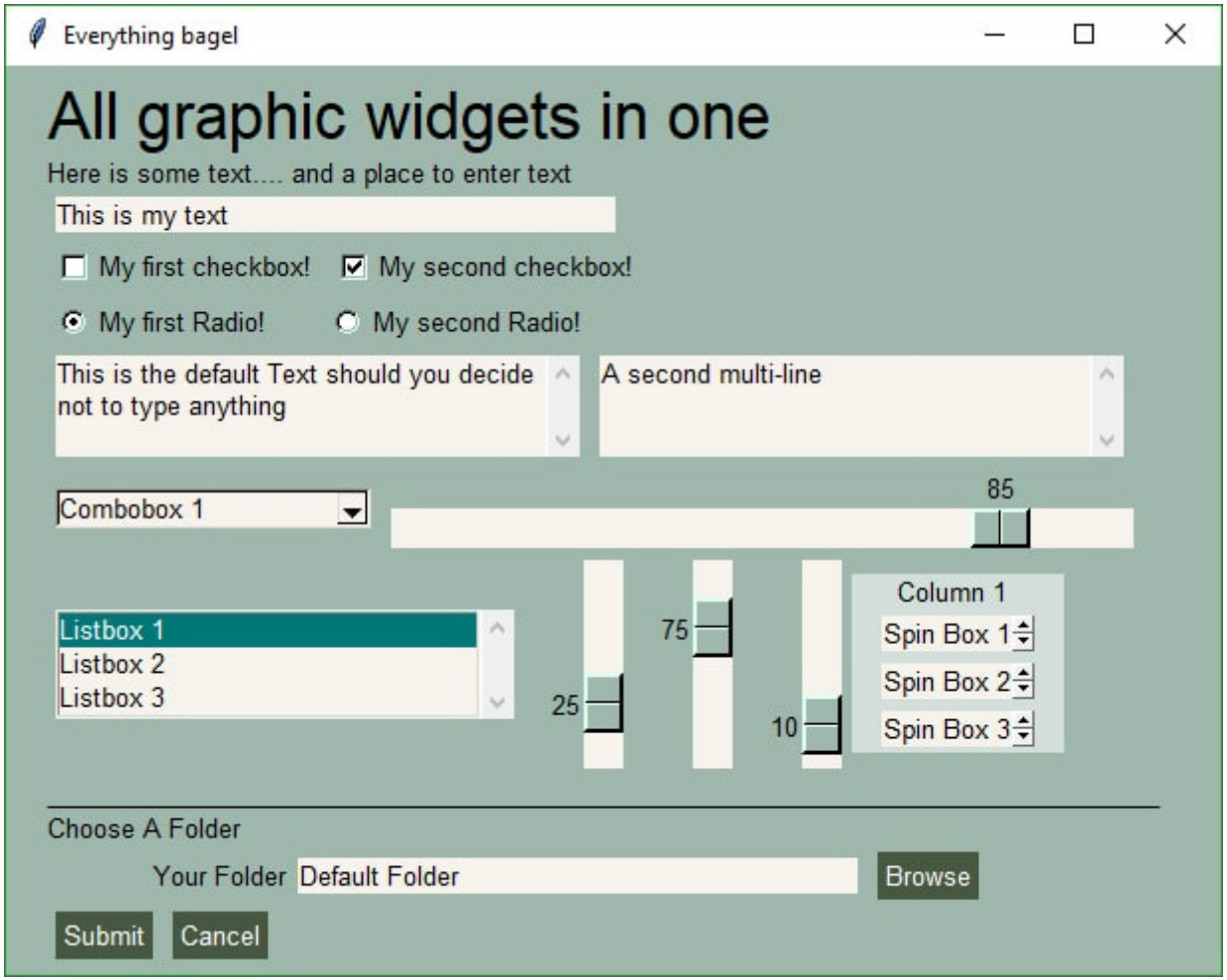
- **Using HTML 5 for GUI in C++**
- Three approaches and many **variations**
- More **inspirational** than educational
    - Consider these approaches
    - Maybe use them
- Some examples to help you **start experimenting** right away

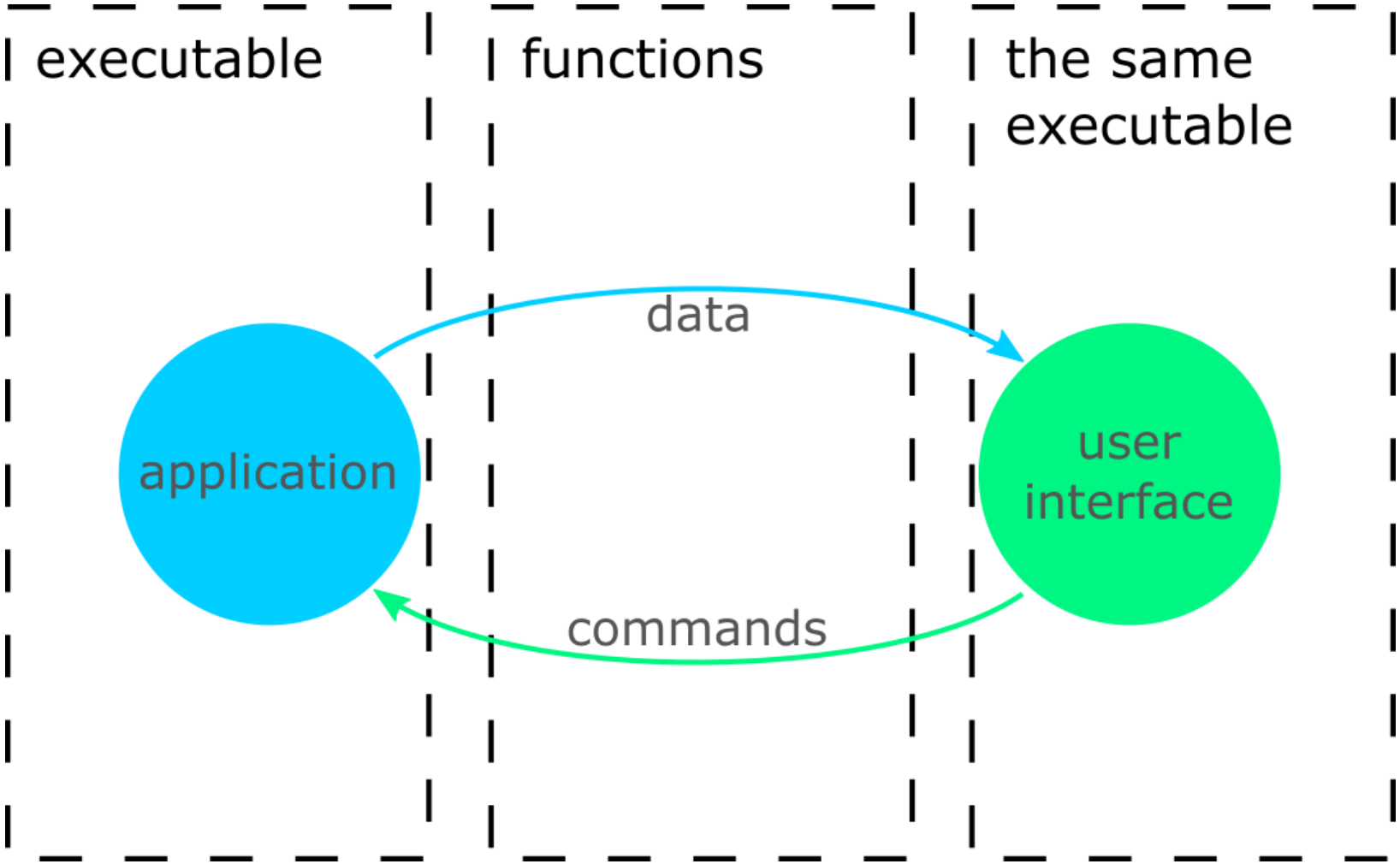
So you want to make a GUI?

Typical user interface



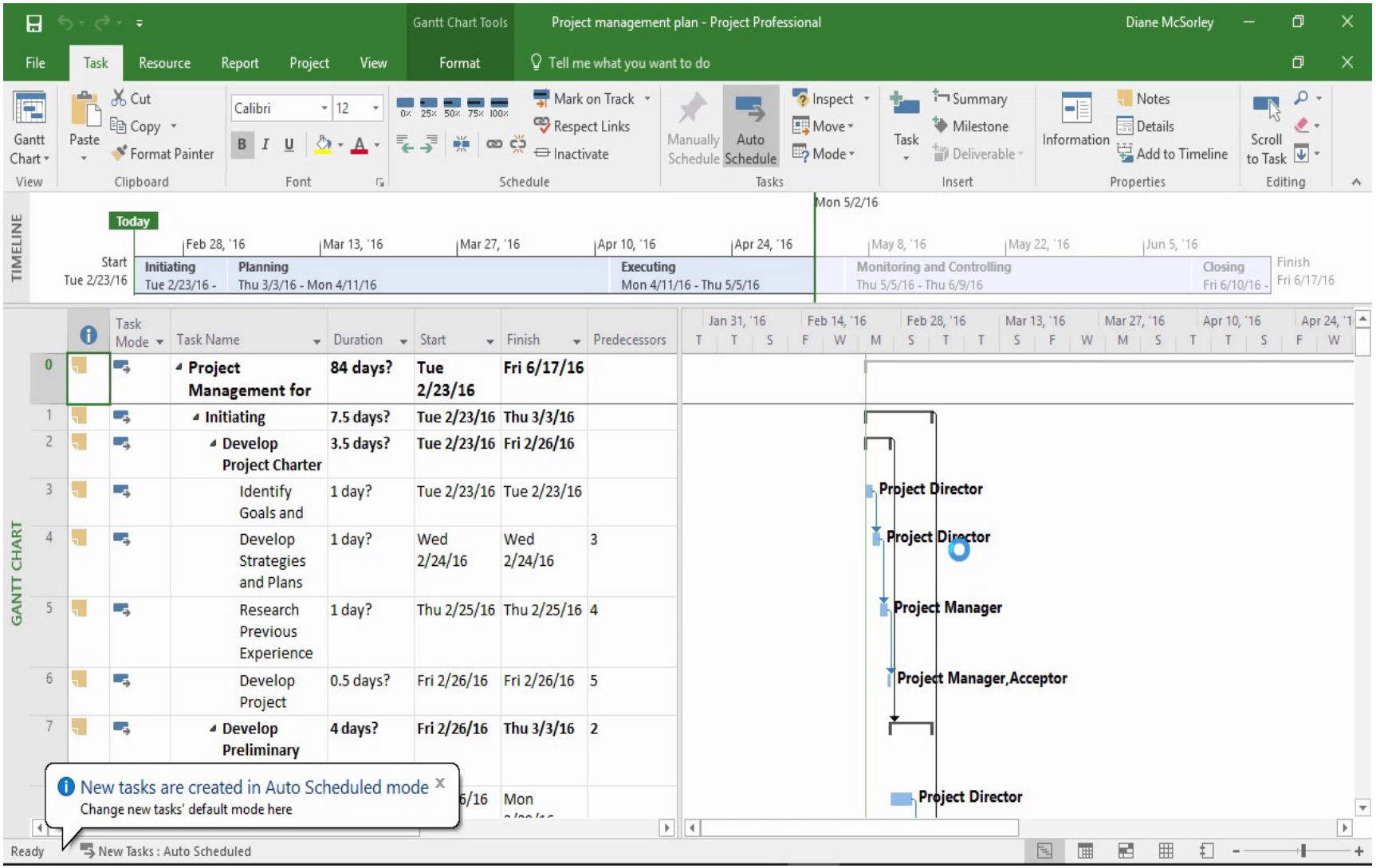
Simple


Everything bagel
All graphic widgets in one
Here is some text.... and a place to enter text
This is my text
My first checkbox!
My second checkbox!
My first Radio!
My second Radio!
This is the default Text should you decide not to type anything
A second multi-line
Combobox 1
85
Listbox 1
Listbox 2
Listbox 3
25
75
10
Column 1
Spin Box 1
Spin Box 2
Spin Box 3
Choose A Folder
Your Folder
Default Folder
Browse
Submit
Cancel

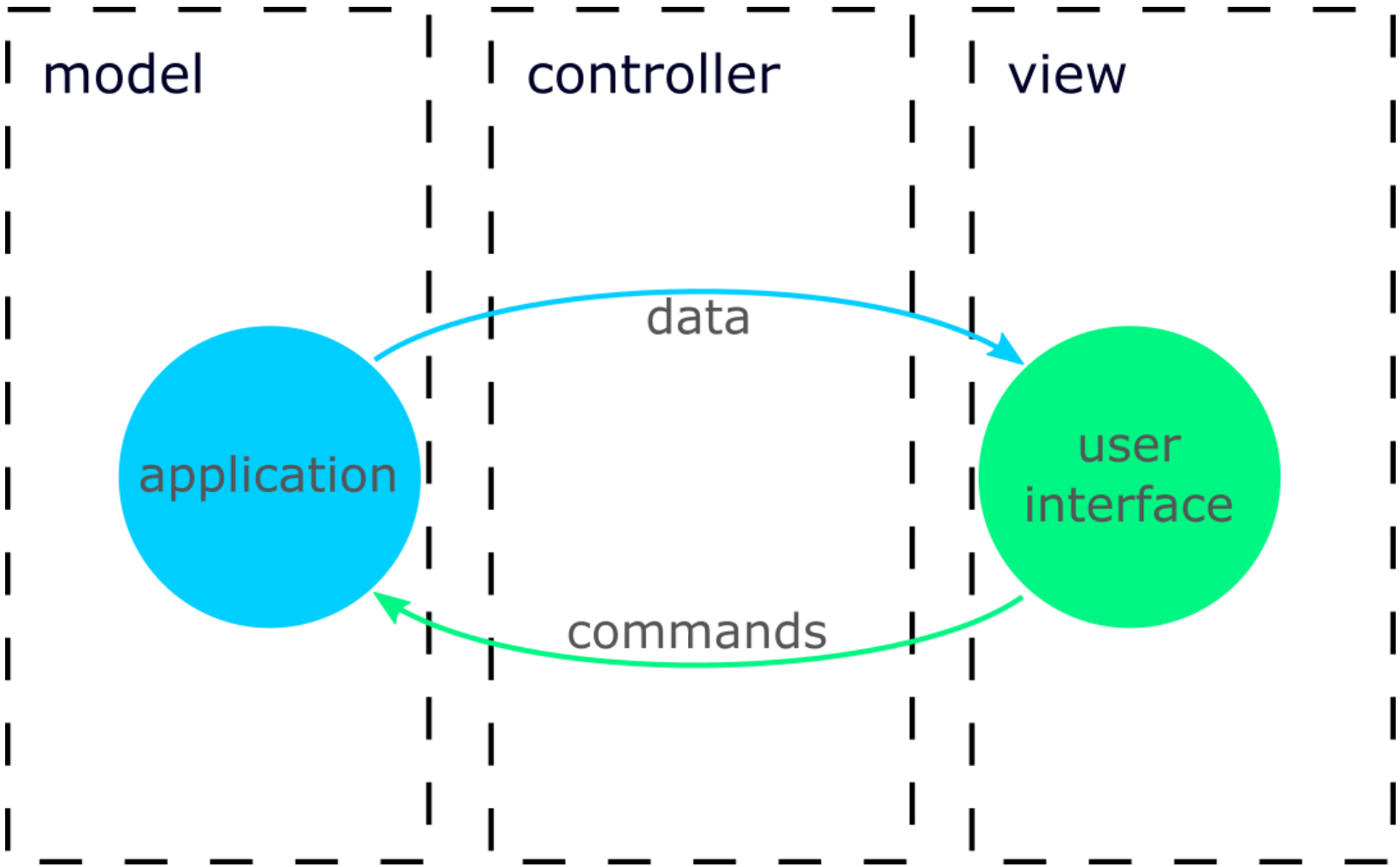

executable
functions
the same executable
application
data
commands
user interface

- GTK
- WxWidgets
- WindowsForms
- ncurses
- MFC
- Many, many more...
- **libui: gh/andlabs/libui**
- **Dear ImGui: gh/ocornut/imgui**

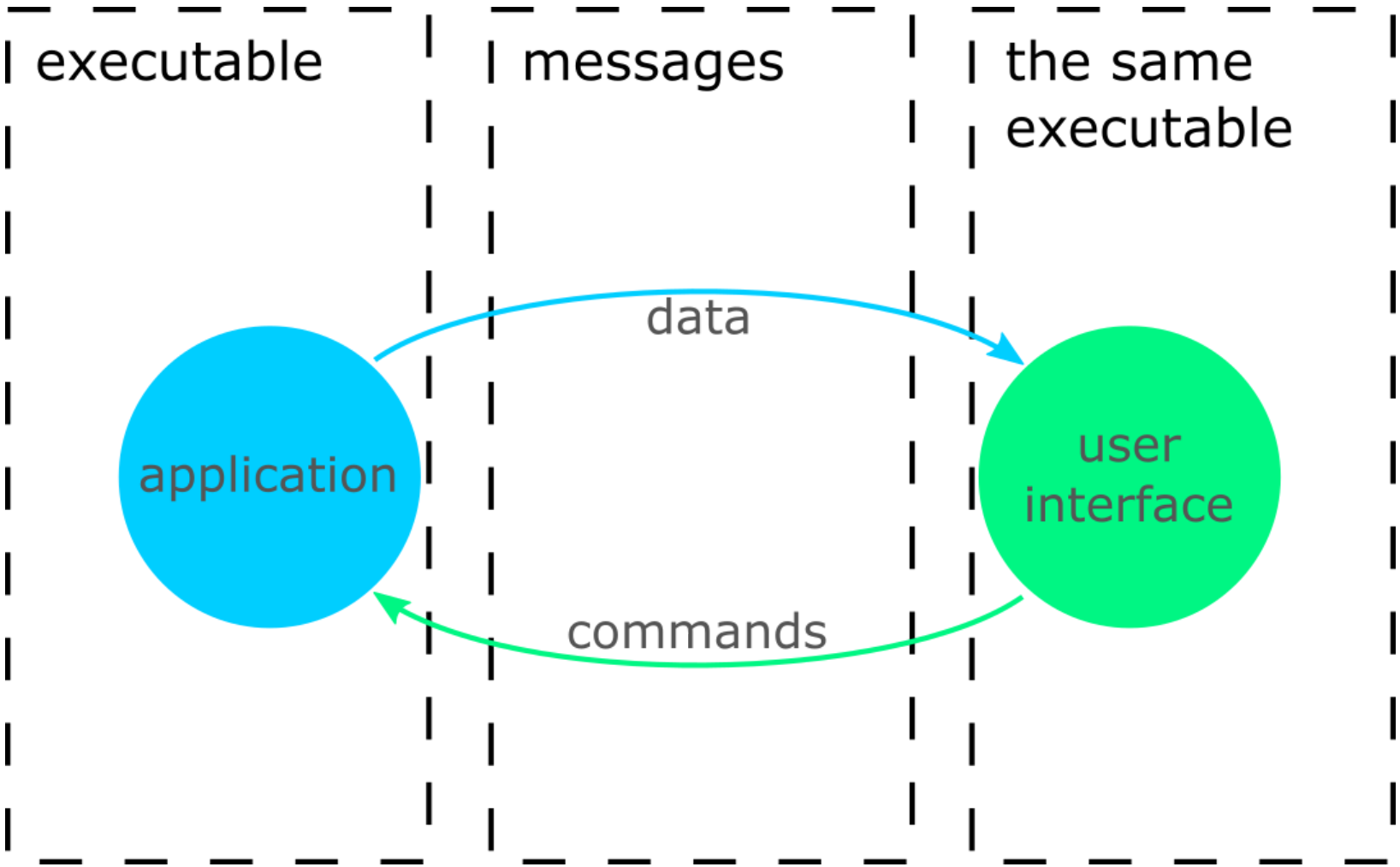
Complex


Gantt Chart Tools
Project management plan - Project Professional
Diane McSorley
File
Task
Resource
Report
Project
View
Format
Tell me what you want to do
Gantt Chart
Paste
Cut
Copy
Format Painter
Calibri
12
Mark on Track
Respect Links
Inactivate
Manually Schedule
Auto Schedule
Inspect
Move
Mode
Task
Summary
Milestone
Deliverable
Information
Notes
Details
Add to Timeline
Scroll to Task
View
Clipboard
Font
Schedule
Tasks
Insert
Properties
Editing
TIMELINE
Today
Mon 5/2/16
Feb 28, '16
Mar 13, '16
Mar 27, '16
Apr 10, '16
Apr 24, '16
May 8, '16
May 22, '16
Jun 5, '16
Start Tue 2/23/16
Initiating Tue 2/23/16 -
Planning Thu 3/3/16 - Mon 4/11/16
Executing Mon 4/11/16 - Thu 5/5/16
Monitoring and Controlling Thu 5/5/16 - Thu 6/9/16
Closing Fri 6/10/16 -
Finish Fri 6/17/16
GANTT CHART
Task Mode
Task Name
Duration
Start
Finish
Predecessors
0 Project Management for 84 days? Tue 2/23/16 Fri 6/17/16
1 Initiating 7.5 days? Tue 2/23/16 Thu 3/3/16
2 Develop Project Charter 3.5 days? Tue 2/23/16 Fri 2/26/16
3 Identify Goals and 1 day? Tue 2/23/16 Tue 2/23/16
4 Develop Strategies and Plans 1 day? Wed 2/24/16 Wed 2/24/16 3
5 Research Previous Experience 1 day? Thu 2/25/16 Thu 2/25/16 4
6 Develop Project 0.5 days? Fri 2/26/16 Fri 2/26/16 5
7 Develop Preliminary 4 days? Fri 2/26/16 Thu 3/3/16 2
Jan 31, '16
Feb 14, '16
Feb 28, '16
Mar 13, '16
Mar 27, '16
Apr 10, '16
Apr 24, '1
Project Director
Project Director
Project Manager
Project Manager,Acceptor
Project Director
New tasks are created in Auto Scheduled mode
Change new tasks' default mode here
Ready
New Tasks : Auto Scheduled

MVC


model
controller
view
data
application
user
interface
commands

Typical C++ MVC




Qt

?

A challenger appears

Browsers


Location: about:
What's New?
What's Cool?
Destinations
Net Search
People
Software
Netscape Navigator™
Version 3.0
Copyright © 1994-1996 Netscape Communications Corporation, All rights reserved.
This software is subject to the license agreement set forth in the license. Please read and agree to all terms before using this software.
Report any problems through the feedback page.
Netscape Communications, Netscape, Netscape Navigator and the Netscape Communications logo are trademarks of Netscape
NETSCAPE

They've come a long way





sophisticated **optimized**

powerful...

This is a browser

Checkbox 1 ☐ Checkbox 2 ☐

Radio 1 ◉ Radio 2 ○ Radio 78 ○

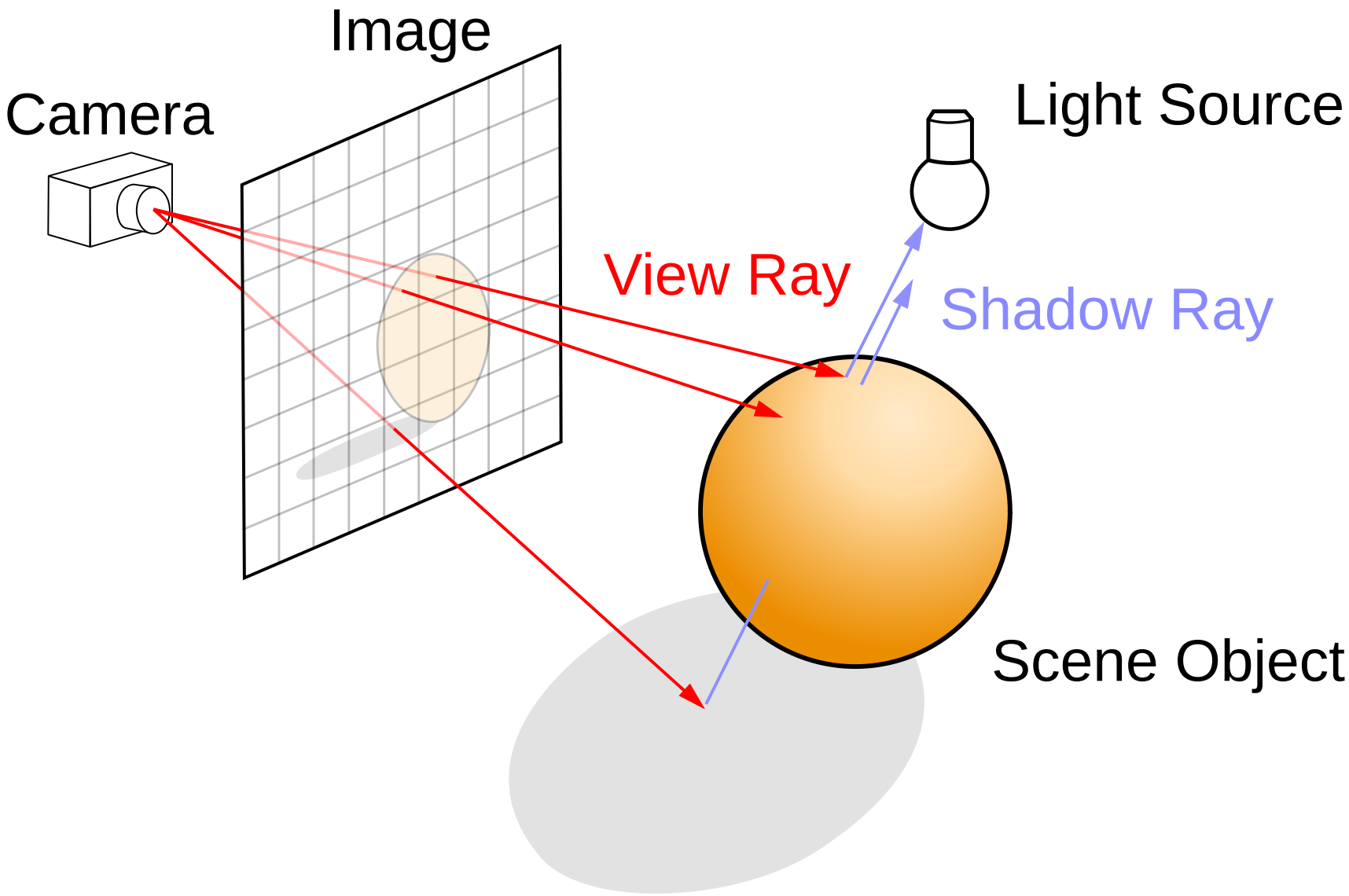


Image
Camera
Light Source
View Ray
Shadow Ray
Scene Object


VAYAGIF.COM

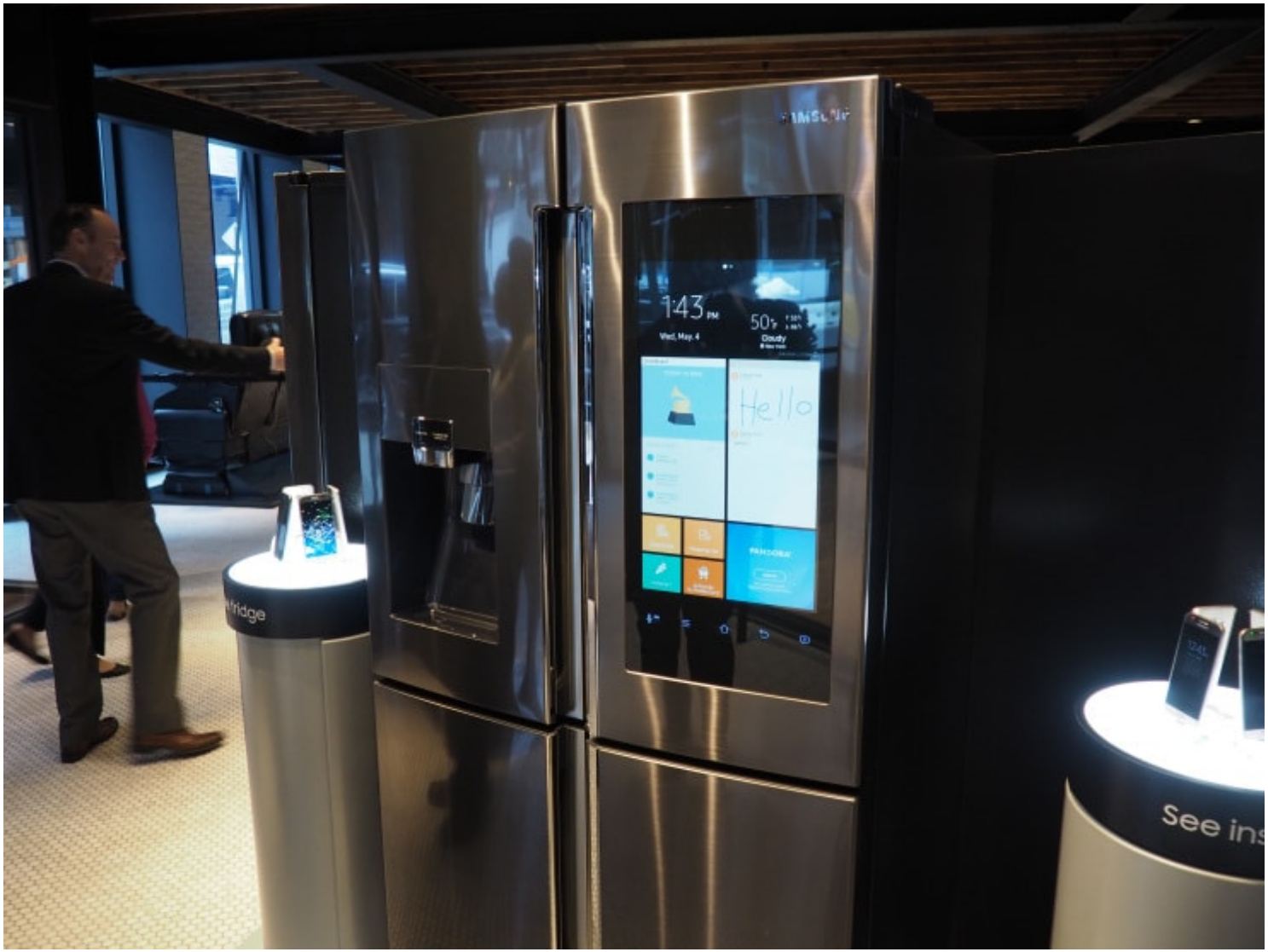
The last 5 slides took me 10 minutes to make

The browser is an **immensely** powerful presentation platform

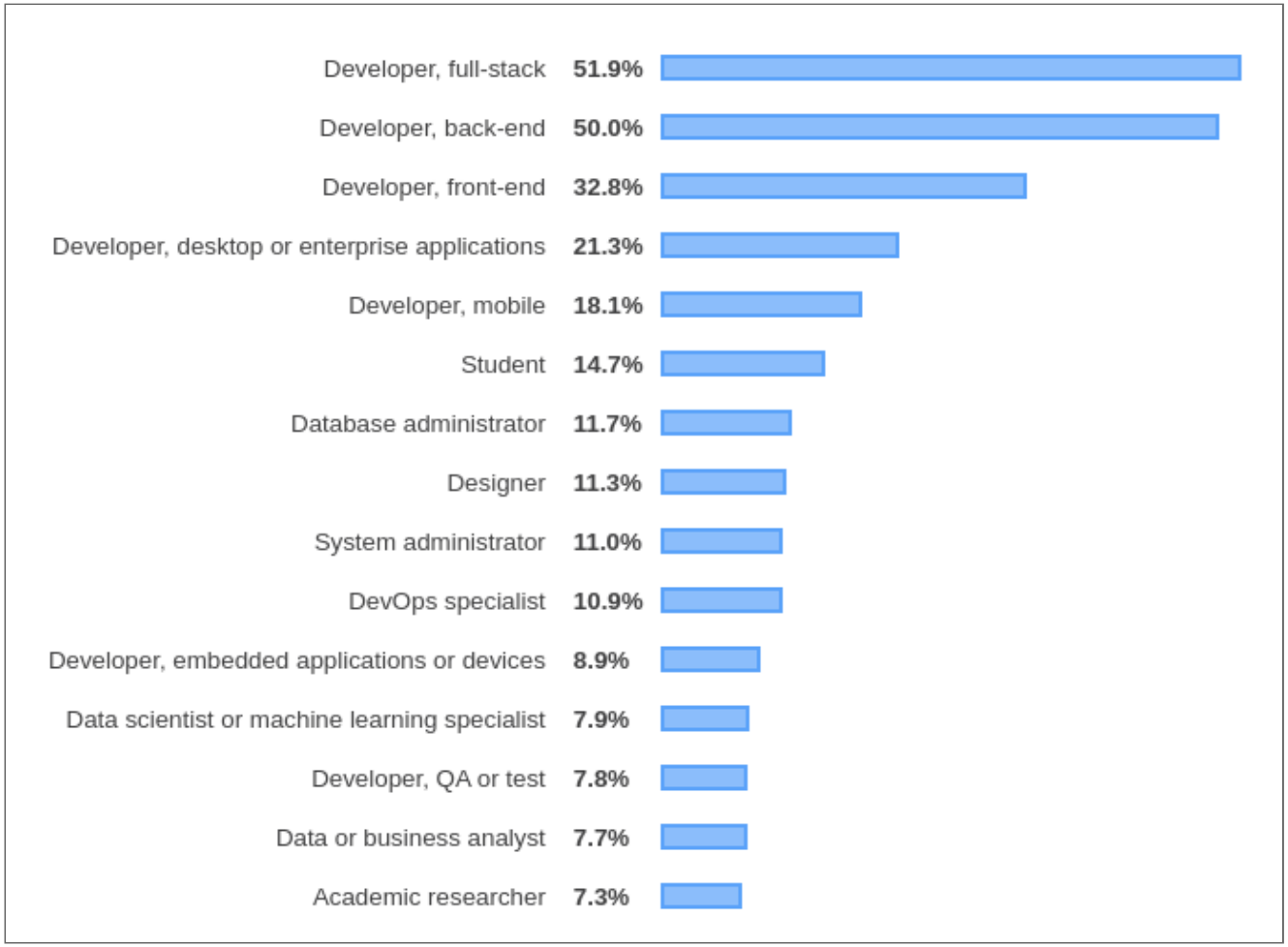
It's our **user interface to the Internet**

sophisticated **optimized**

powerful... **ubiquitous**

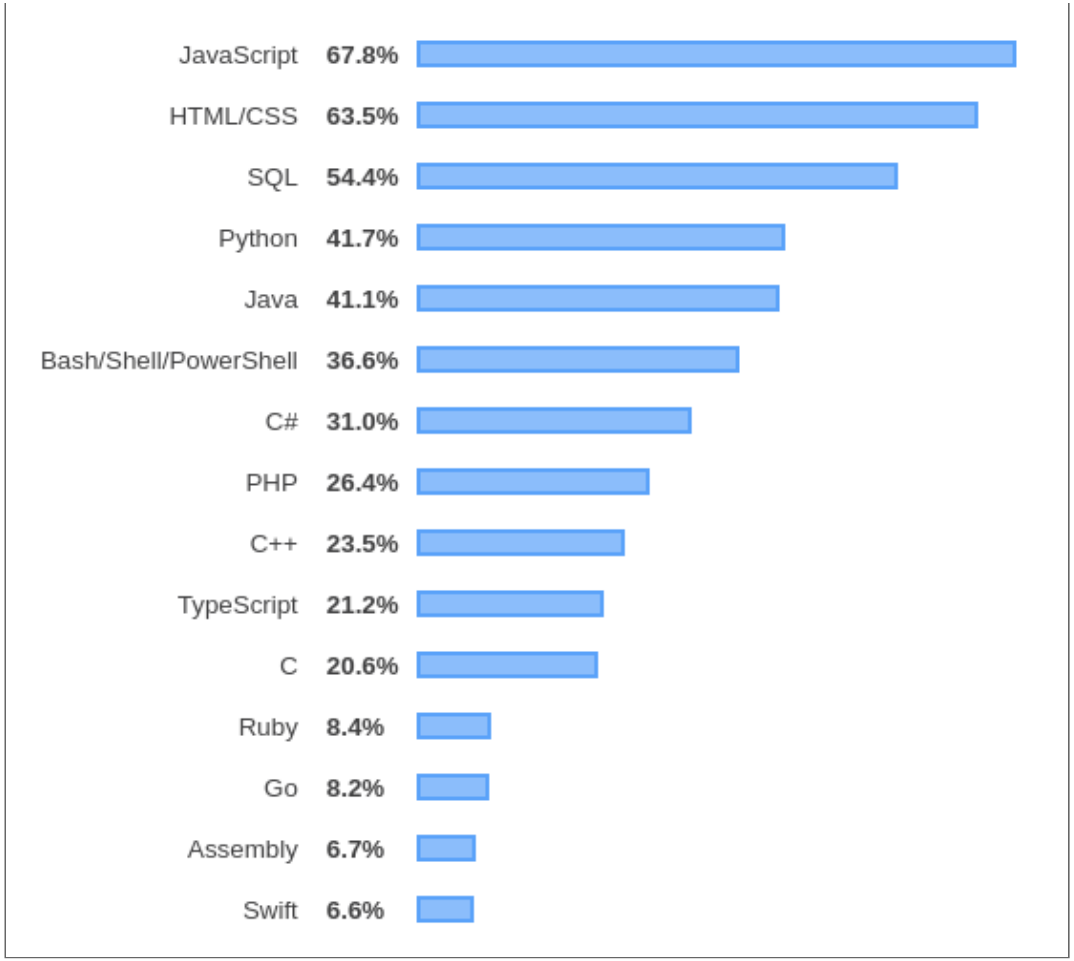

1:43 PM
Hello
See ins

Developer Type

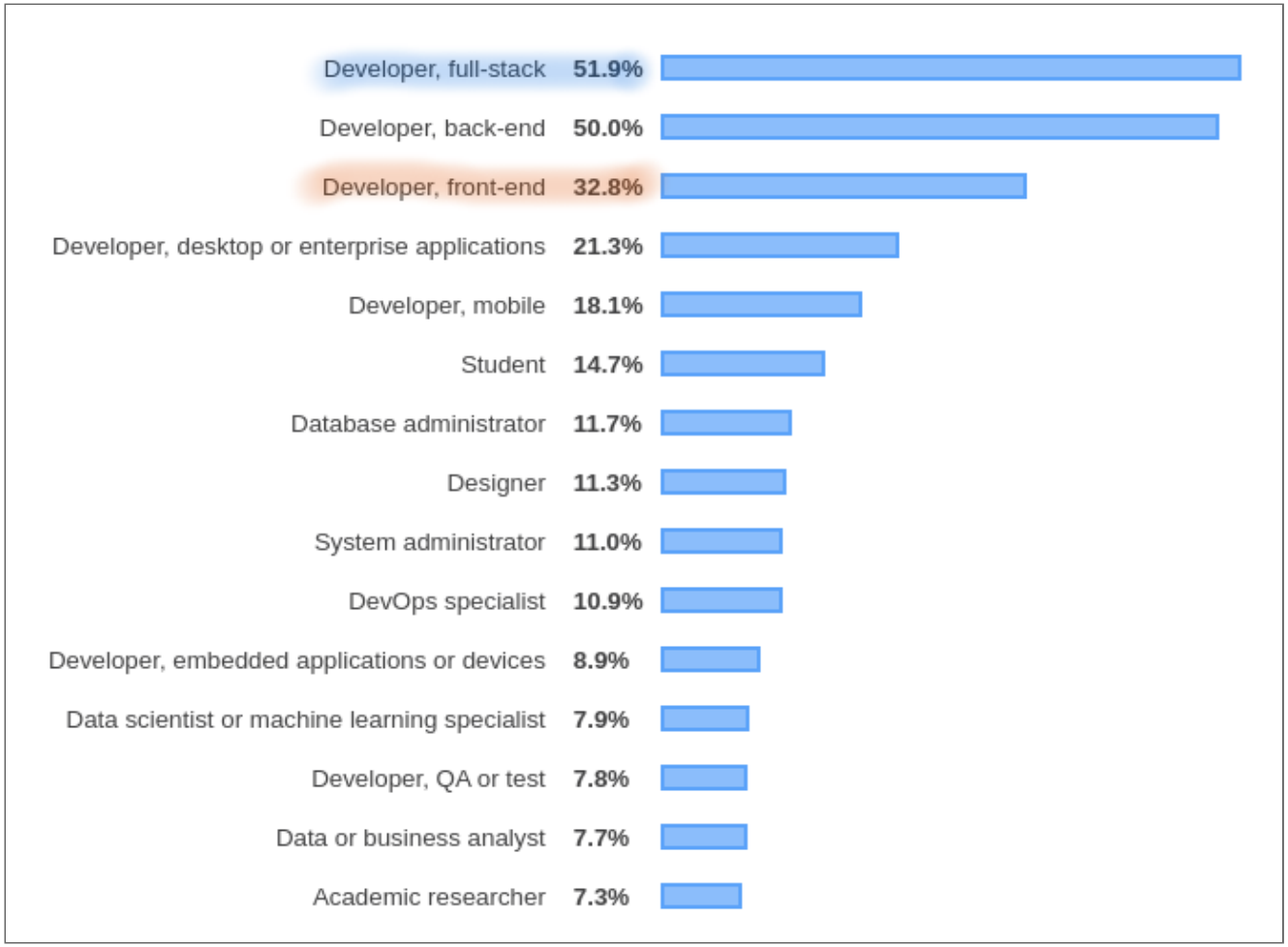


Programming Language

source: **Stack Overflow Survey 2019**

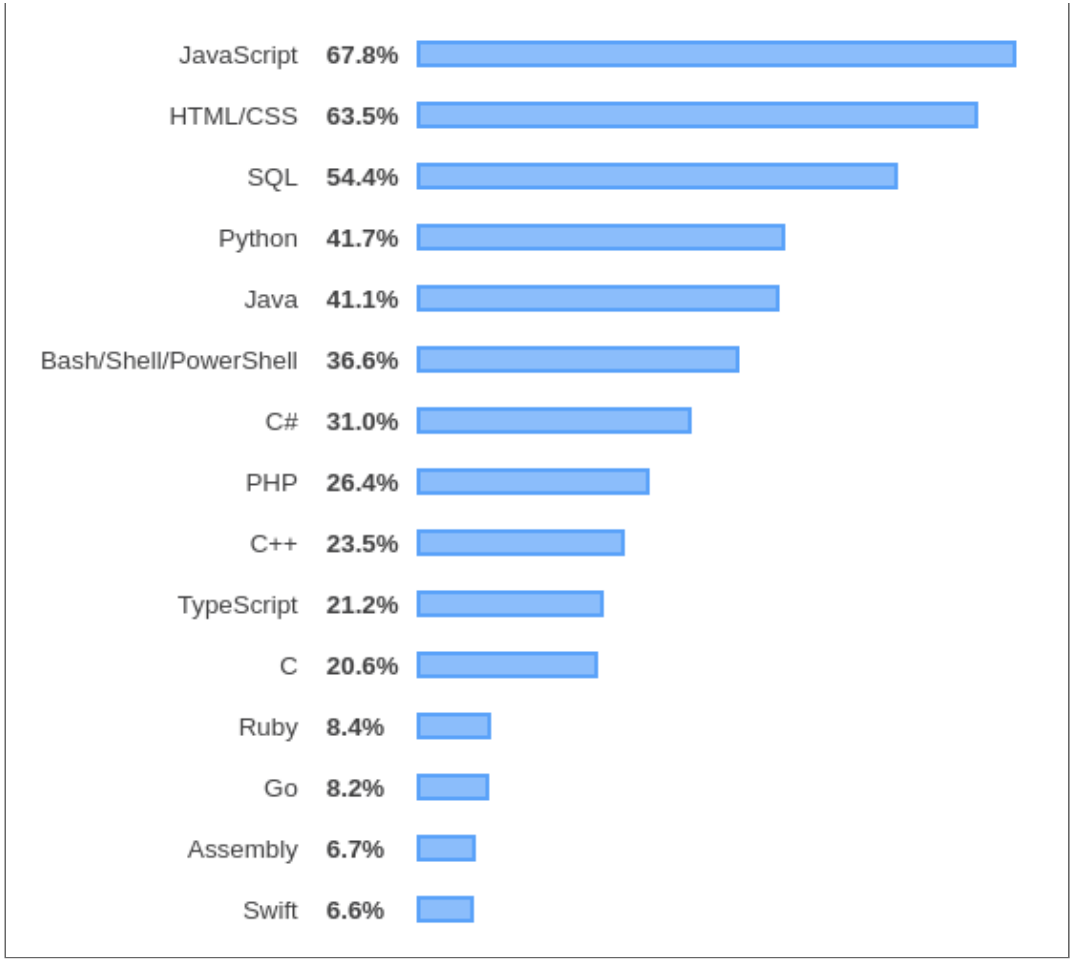

JavaScript 67.8%
HTML/CSS 63.5%
SQL 54.4%
Python 41.7%
Java 41.1%
Bash/Shell/PowerShell 36.6%
C# 31.0%
PHP 26.4%
C++ 23.5%
TypeScript 21.2%
C 20.6%
Ruby 8.4%
Go 8.2%
Assembly 6.7%
Swift 6.6%

Developer Type

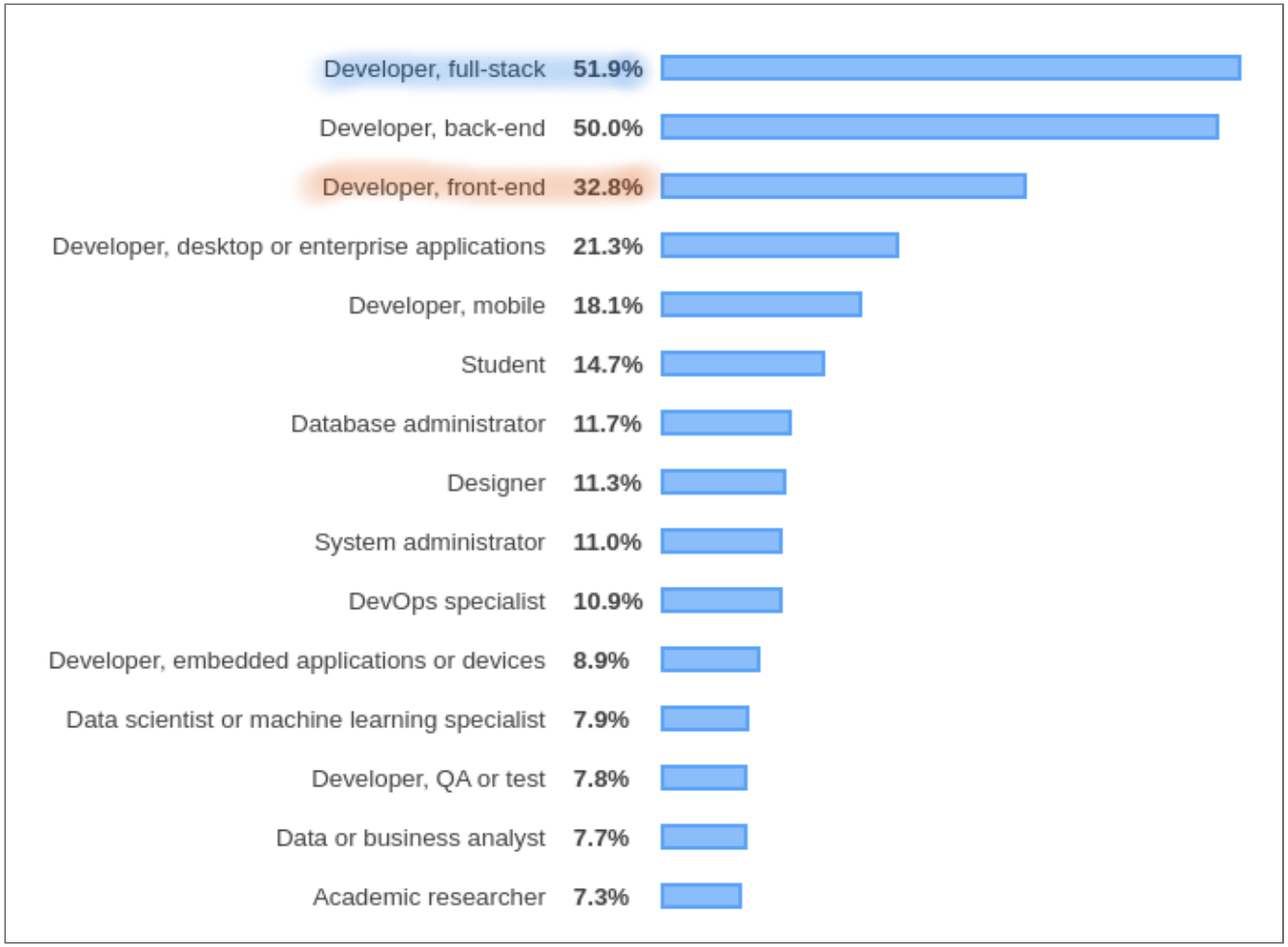


Programming Language

source: **Stack Overflow Survey 2019**

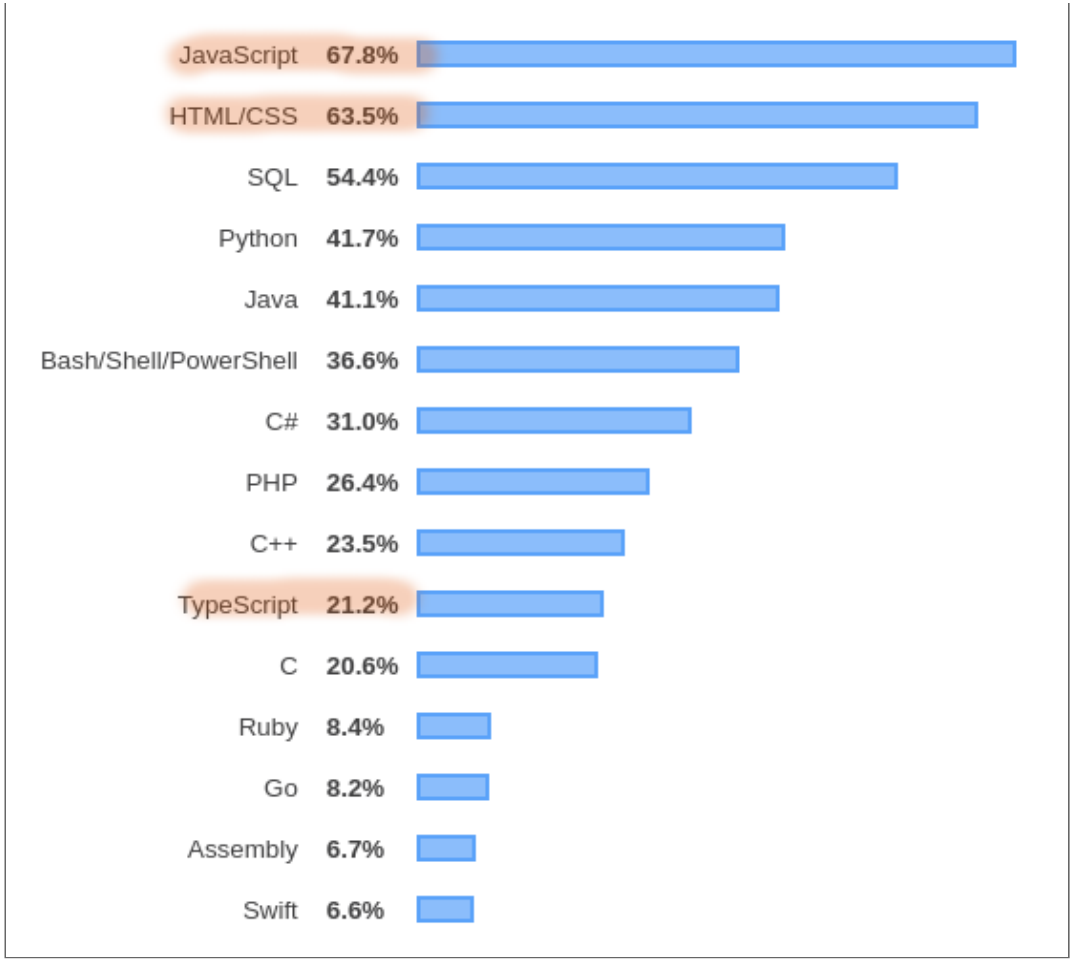

JavaScript 67.8%
HTML/CSS 63.5%
SQL 54.4%
Python 41.7%
Java 41.1%
Bash/Shell/PowerShell 36.6%
C# 31.0%
PHP 26.4%
C++ 23.5%
TypeScript 21.2%
C 20.6%
Ruby 8.4%
Go 8.2%
Assembly 6.7%
Swift 6.6%

Developer Type



Programming Language

source: **Stack Overflow Survey 2019**


JavaScript 67.8%
HTML/CSS 63.5%
SQL 54.4%
Python 41.7%
Java 41.1%
Bash/Shell/PowerShell 36.6%
C# 31.0%
PHP 26.4%
C++ 23.5%
TypeScript 21.2%
C 20.6%
Ruby 8.4%
Go 8.2%
Assembly 6.7%
Swift 6.6%

Finding experienced and competent HTML 5 developers is easier than finding experienced and competent C++ developers

Can we make use of all that?

What is HTML 5?

Briefly

HTML 5 in a single slide

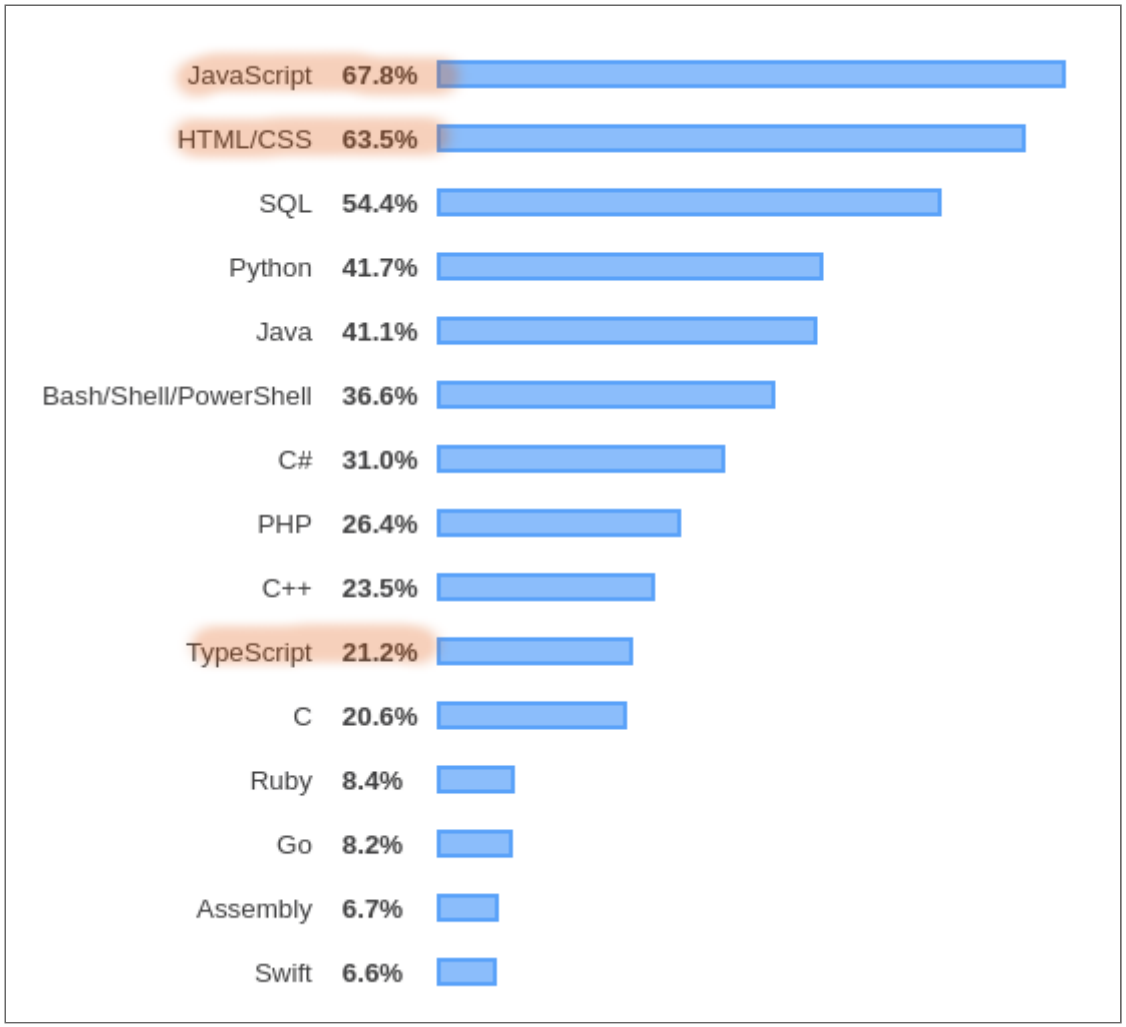
- It's not a single thing
- A stack of **four languages**
- Declarative languages **define the DOM**
  - **HTML** (including SVG) - defines elements
  - **CSS** - defines the style (appearance) of elements
- Imperative languages **modify the DOM**
  - **JavaScript**
  - **WebAssembly**

That's what every single web page is

Including this one

What is HTML 5 development?

More than simply writing HTML, CSS, and JavaScript



That's a huge community

Languages

- CSS (a terrible language)
  - Hardly anyone writes pure CSS
  - **Less, Sass**, many more alternatives
- JavaScript (a quite decent language)
  - Many people use other languages
  - **TypeScript, CoffeeScript**, and many more
  - Asm.js and C++
- **Many languages** for HTML and WebAssembly

Workflows

- **node.js** - it's for the front end too
- Package managers: **npm, yarn, webpack...**
- `$ npm start` - the ultimate build server
- Debugging: Dev Tools. **F12**
- Frameworks: **React, Vue, Angular**, and more

HTML 5 Development Is Modern

Approach 1

A local HTTP server

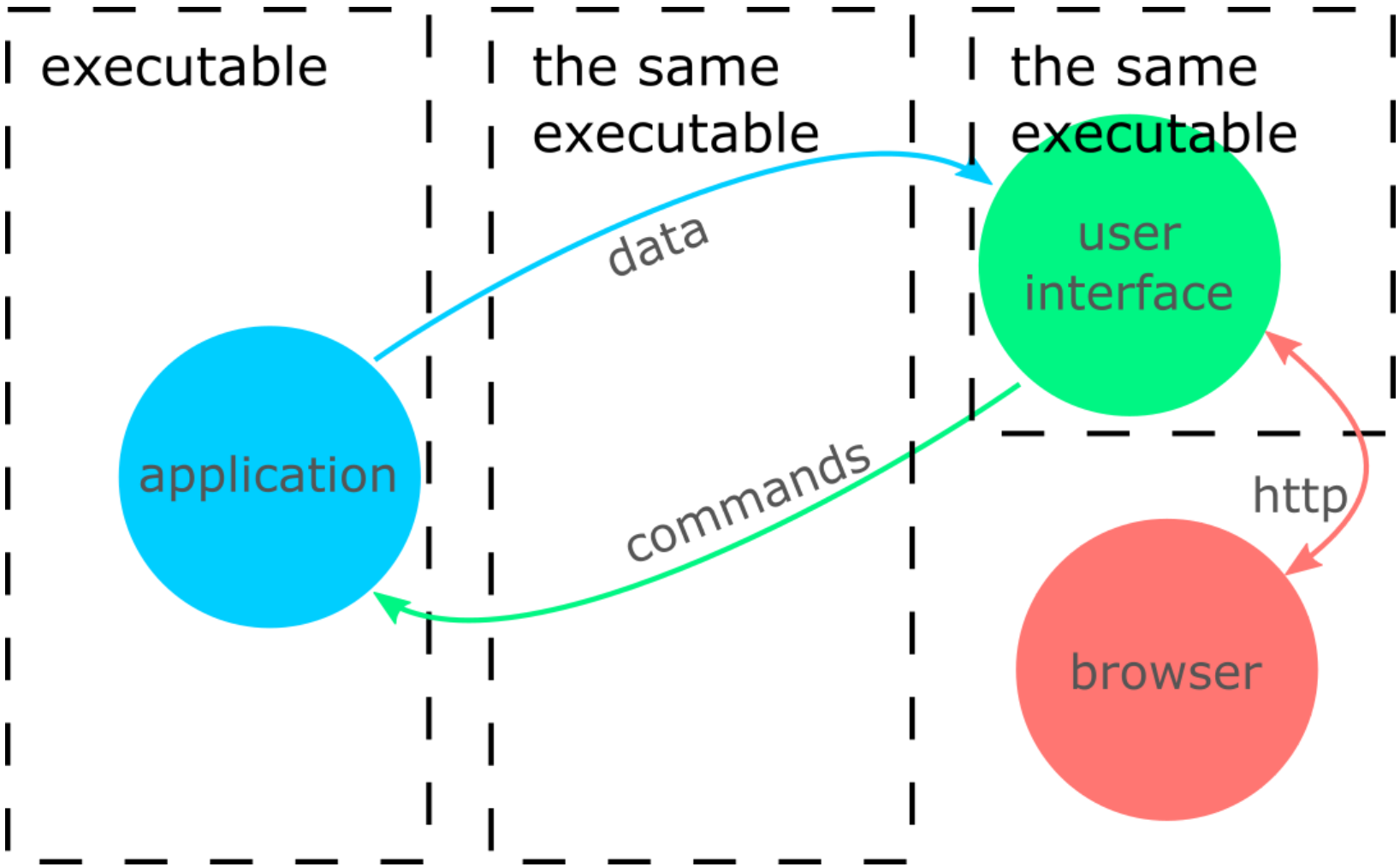
1. Start HTTP server - our C++ application
2. Open browser to **"http://localhost:1234"**

Done since the 1990's

Multi-Page Apps

- Click hyperlink. Request page. Server generates it and sends it.
- `$ python -m SimpleHTTPServer 8080`
- Serving dynamic content = **GUI code on the server**

Multi-page architecture



- Multi-page apps are **not interactive**
- They are generally a **bad idea**
- They can qualify as a **simple GUI**
- The PHP programmer's GUI of choice

Single-Page Apps

- JavaScript modifies the DOM
- No new URL on every click
- Welcome to 2004: Ajax and XHR
- `XMLHttpRequest`. It's not for XML. Any text or binary works
- **Not very interactive.** Everything needs to be polled
- The interactivity **hack**

An Ajax HXR loop

```
function idle() {
    fetch('clientReady') // get some "fake" service resource
        .then(data => {
            dispatch(xhr.responseText); // dispatch and use the response
            idle(); // request more from the server
        })
        .catch(error => console.log('Error: ' + xhr.status));
}
// ignore other xhr responses
```

But where is `XMLHttpRequest`?

The 2004 Ajax HXR loop

```
function idle() {
    let xhr = new XMLHttpRequest();
    xhr.open('GET', 'clientReady'); // get some "fake" service resource
    xhr.onreadystatechange = function () {
        if (xhr.readyState === 4) { // 4 = request done
            if (xhr.status === 200) { // 200 = OK
                dispatch(xhr.responseText); // dispatch and use the response
                idle(); // request more from the server
            } else {
                console.log('Error: ' + xhr.status); // error
            }
        }
    };
    xhr.send(null);
}
// ignore other xhr responses
```

PS Do use `fetch` if you do Ajax

Modern Single-Page Apps

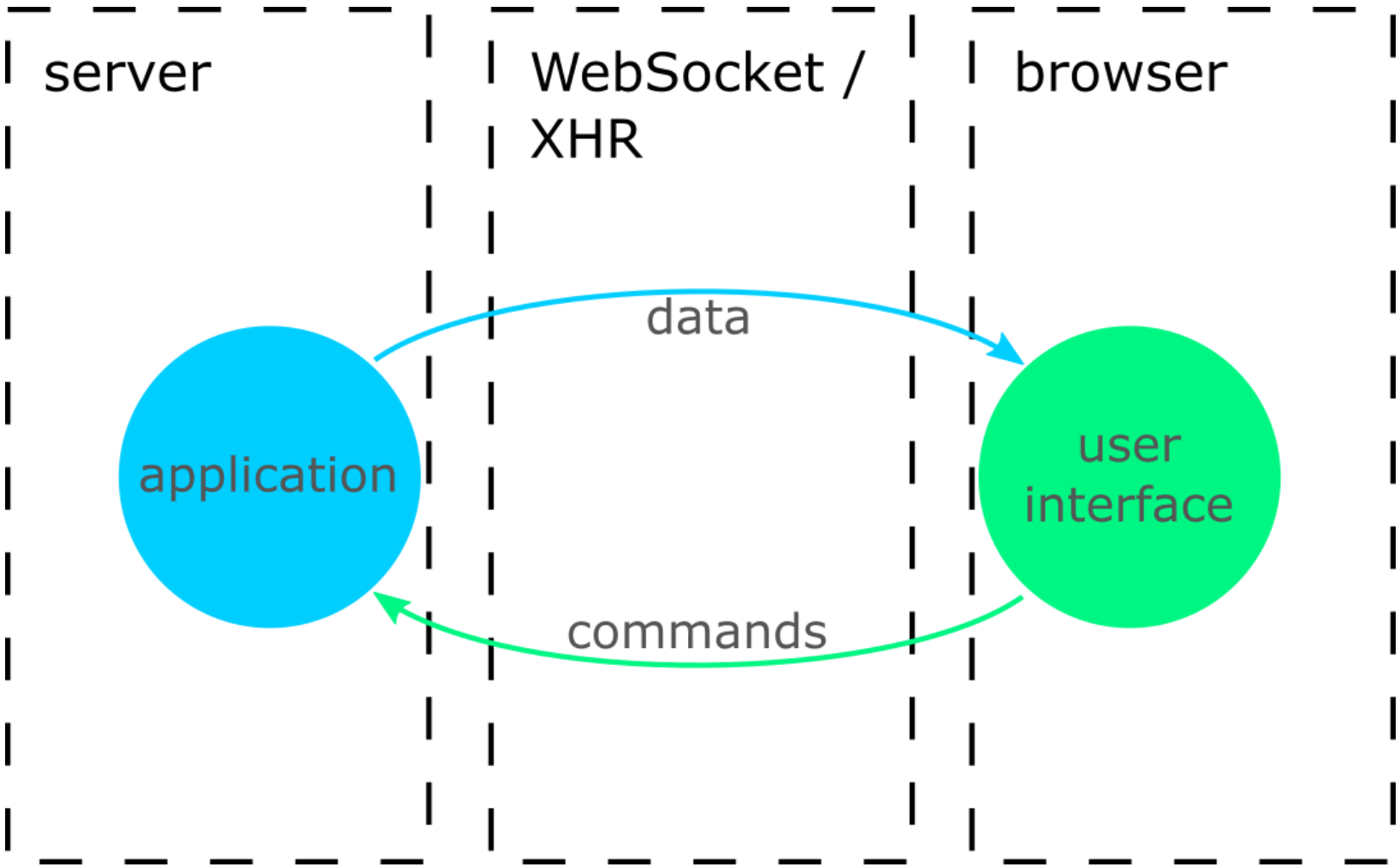
- Welcome to HTML 5: **WebSocket**
- The XHR loop to the next level
- **Simpler and cleaner** communication API
- **More powerful**, too

The WebSocket loop

```
let ws = new WebSocket('ws://localhost:7658');
ws.onopen = myOnConnectionOpenedFunc;
ws.onclose = myOnConnectionClosedFunc;
ws.onmessage = myOnMessageFunc;
ws.onerror = myOnErrorFunc;
...
ws.send('message to the server');
```

A **continuous** connection between the server and the client

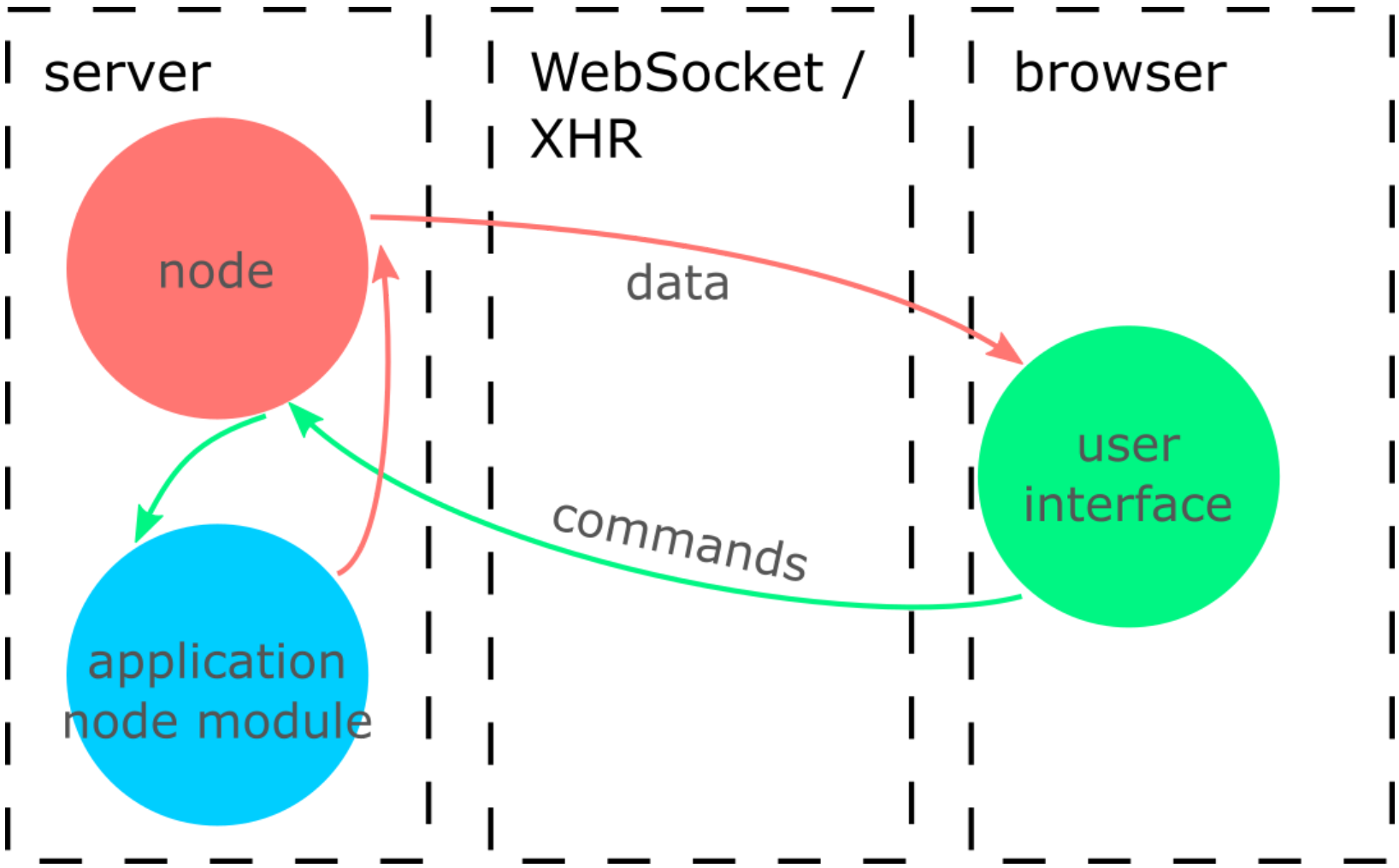
Local HTTP server GUI architecture



But...

- Remember `$ npm start` and the marvelous workflow
- The server is our C++ app. It needs to send HTML 5 content
- We need to reimplement npm? **No**
- HTML 5 developers need to give up their workflow? **No**
- Instead we could...

Node Modules?

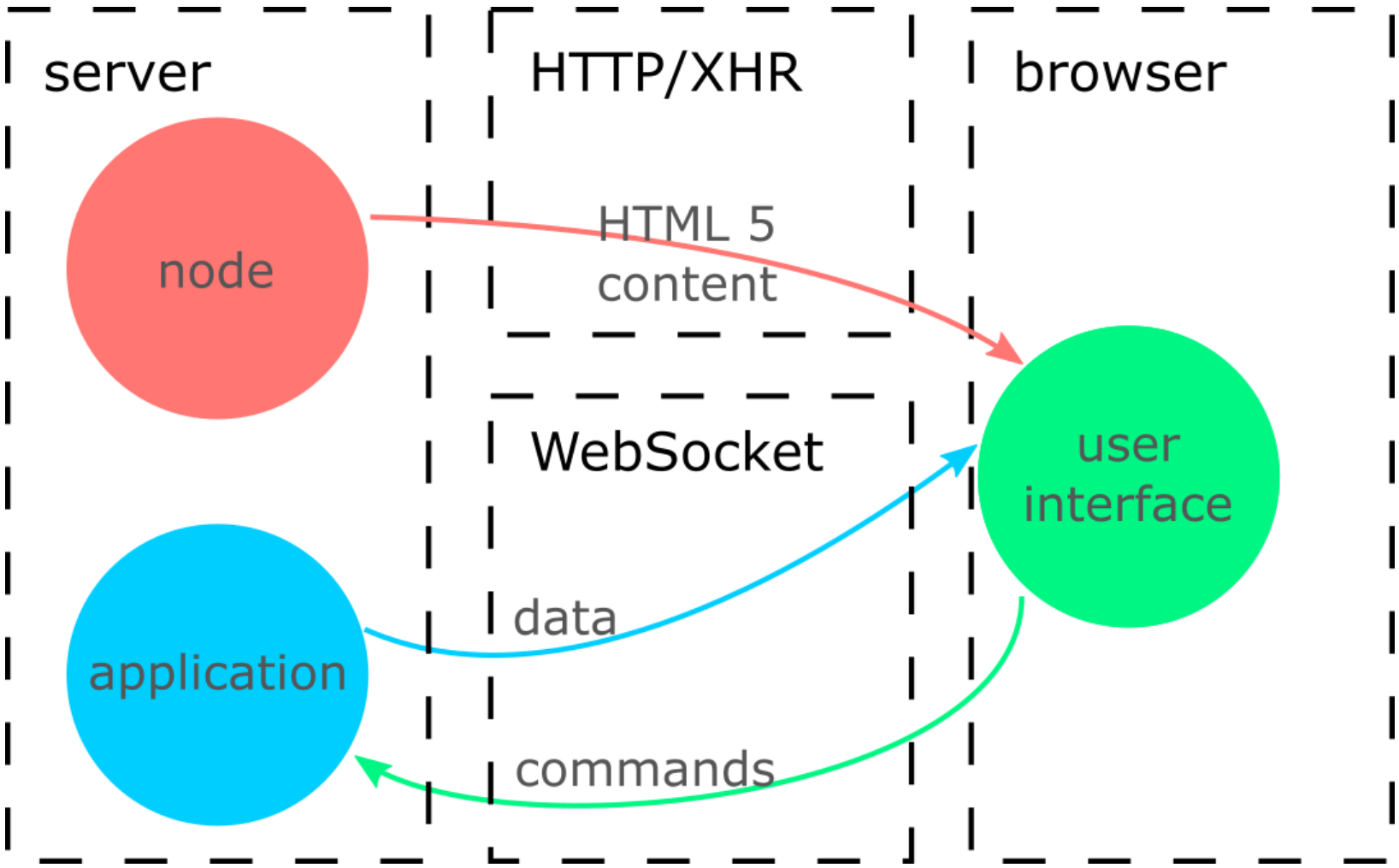


WebSockets are more powerful

```
let ws = new WebSocket('ws://localhost:7658');
ws.onopen = myOnConnectionOpenedFunc;
ws.onclose = myOnConnectionClosedFunc;
ws.onmessage = myOnMessageFunc;
ws.onerror = myOnErrorFunc;
...
ws.send('message to the server');
```

This doesn't have to be the same URL as the HTTP server

WebSocket Architecture



Adding WebSockets to a C++ app

- Many options
- This is CppCon, so I'll use Boost.Beast
- **Echo demo**
- **Boost.Beast docs**
- **CppCon 2016: Vinnie Falco "Introducing Beast..."**
- **CppCon 2018: Vinnie Falco "Get rich quick! Using Boost.Beast WebSockets and Networking TS"**

WebSocket libraries for C++

- Boost.Beast **gh/boostorg/beast** ws/http server/client
- uWebSockets **gh/uNetworking/uWebSockets** ws/http server/client
- WebSockets++ **gh/zaphoyd/websocketpp** ws-only server/client
- Simple-WebSocket-Server **gitlab/eidheim/Simple-WebSocket-Server** ws-only server/client
- Many more: Qt, Poco...

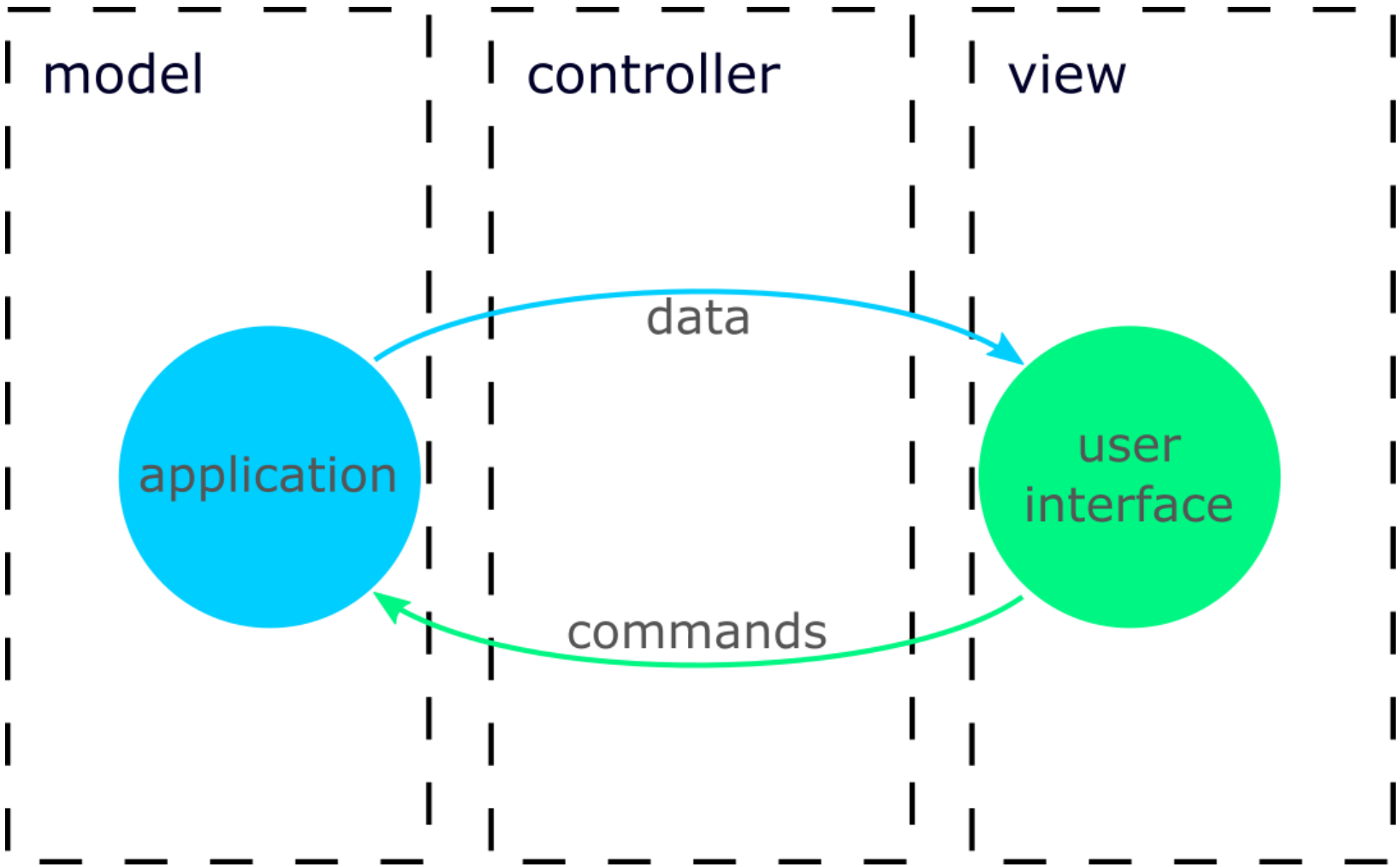
HTTP server pros

- GUI for apps on a virtual machine
  - Virtual Linux with no X Window Server **demo**
- Multiple sessions
  - ... as long as your app supports them
- You don't *have* to use "localhost"
  - Actual **web apps**

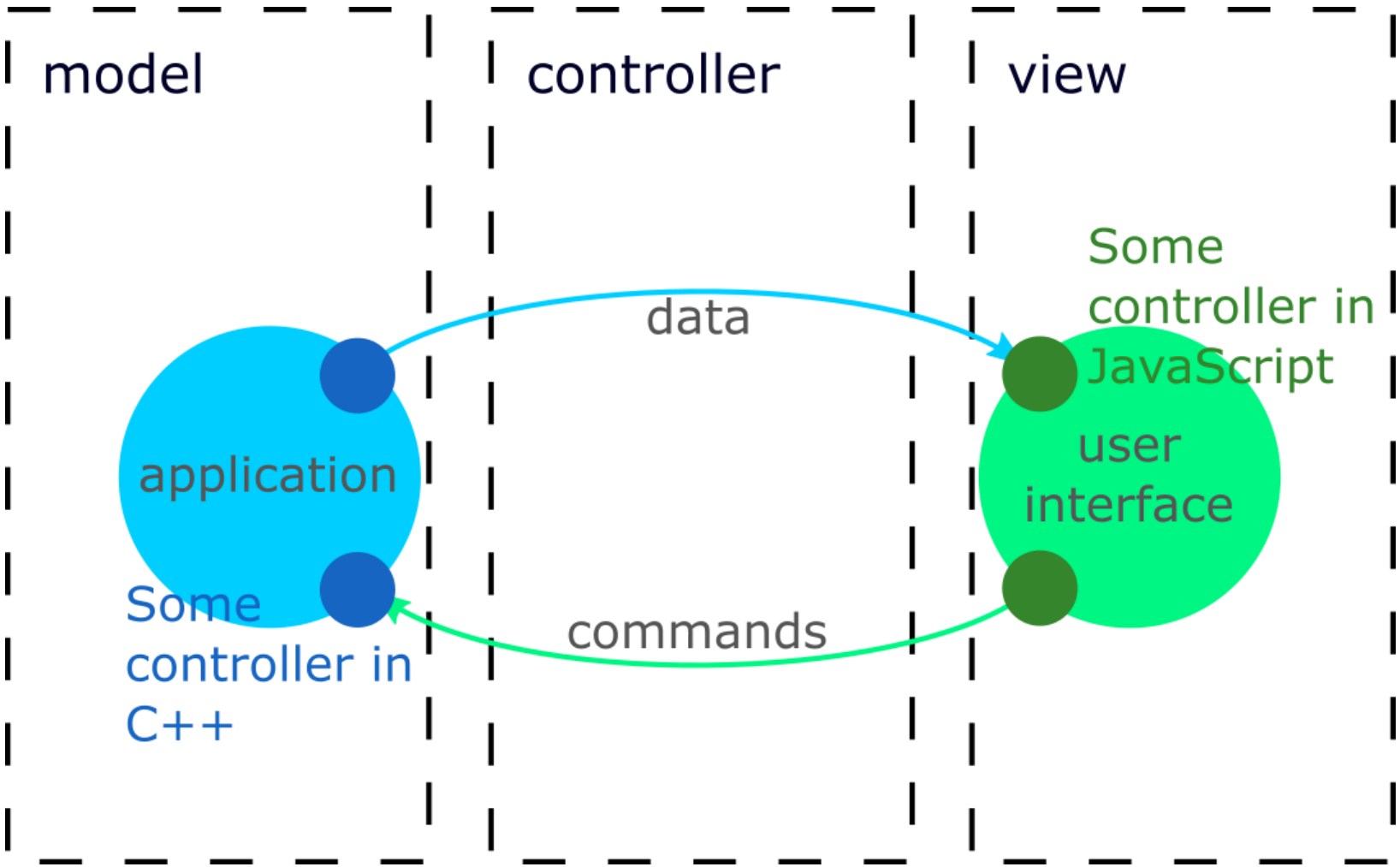
HTTP server cons

- The client is a browser

MVC


model
controller
view
application
data
commands
user
interface

Controller in two languages



HTTP server cons

- The client is a browser
  - Compile C++ to JS or WebAssembly?
- Not a single application: **server(s) and a browser**
- **Lots of load** on the network interface
  - Everything goes through HTTP
  - This might not be a problem on localhost

Approach 2

Embedding a browser

A browser as a library

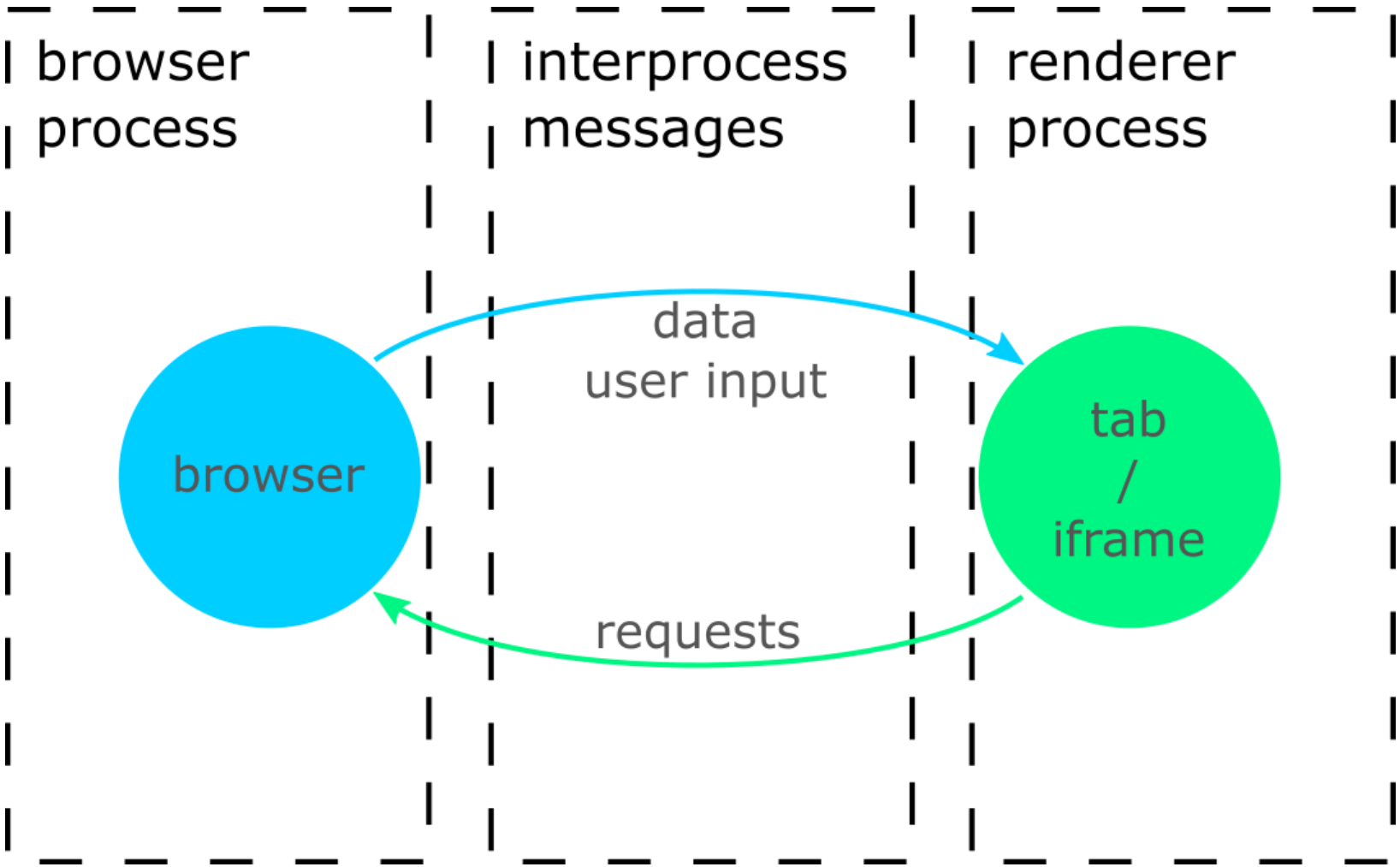
```
openBrowserWindow(url)
```

For many platforms this is `WebView`

Browser architecture in a single slide

- Tabs and iframes
- Potential security issues
- Separate processes - the renderer processes
- Renderer processes are sandboxed
- State between processes = cookies

Browser Architecture



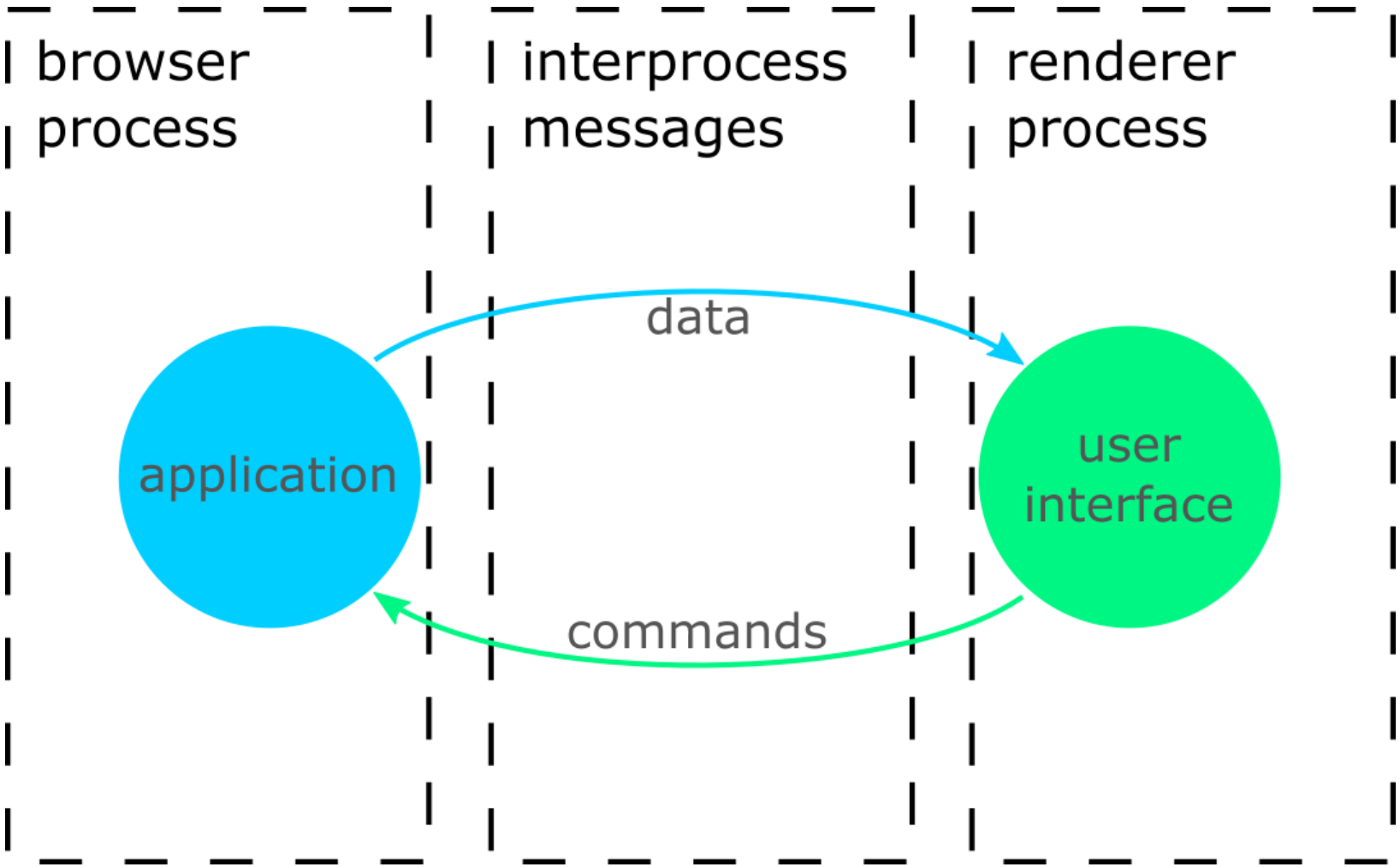
Things to consider when wrapping a browser

BRIB

- Writing code in the **B**rowser process
- Writing code in the **R**enderer process
- **I**nterprocess communication
- C++⇔JS **B**inding

Then it's like an HTTP server

Embedded Browser Architecture



Browser embedding in C++

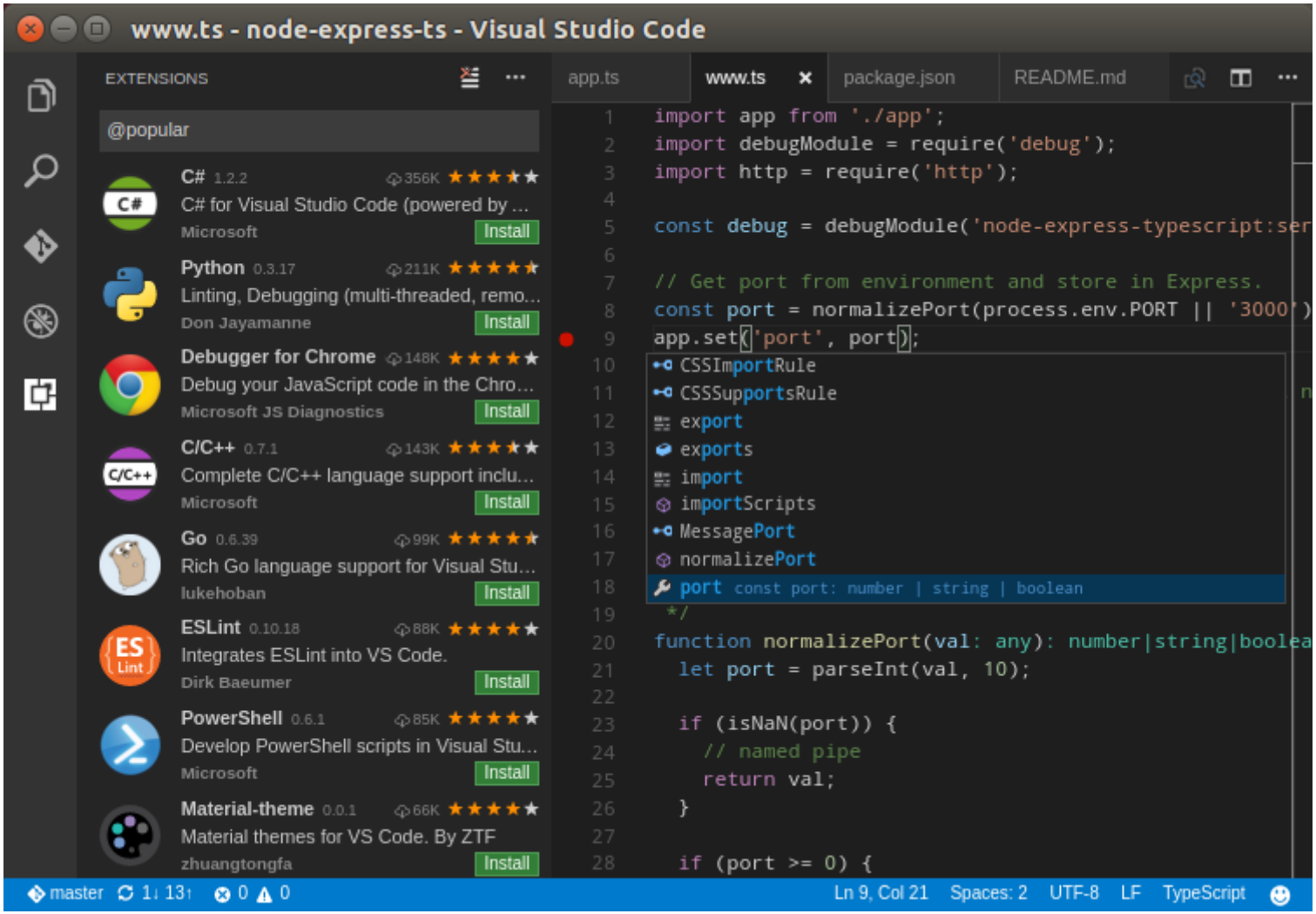
- `QWebView` - wraps native browser.

Browser embedding in C++

- ~~`QWebView`~~ - wraps native browser. **No BRIB**
- **Qt WebEngine** - wraps Chromium
- **CEF** (The Chromium Embedded Framework) - wraps Chromium
- **Electron** - wraps Chromium
- **WebKit**
- Qt WebKit - wraps WebKit. **Outdated**
- Gecko - ???
- **WG21 p1108** - `std::web_view` **Too early to say**

  Chromium is pretty popular.

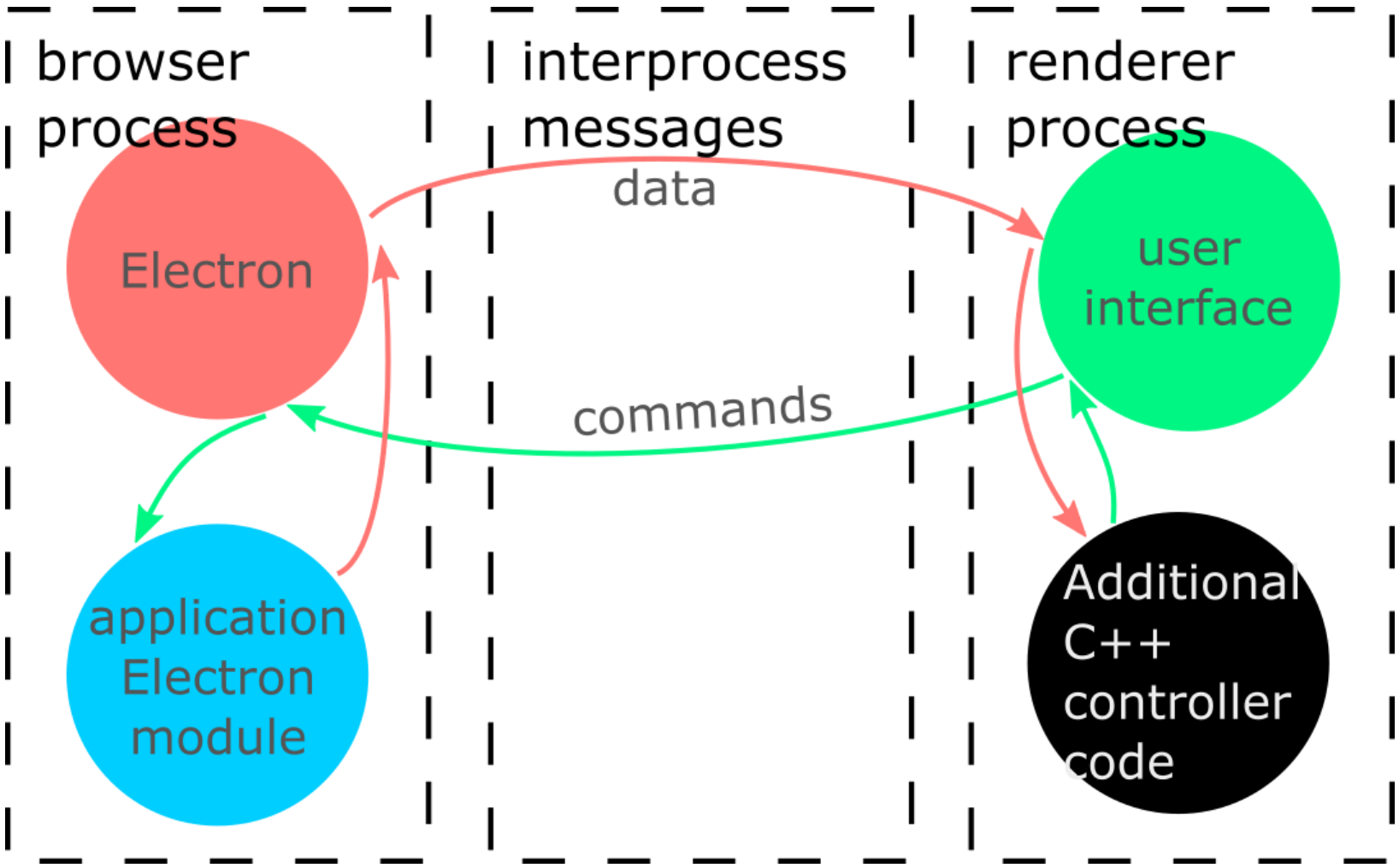
Electron


www.ts - node-express-ts - Visual Studio Code
EXTENSIONS
@popular
C# 1.2.2 356K
C# for Visual Studio Code (powered by ...
Microsoft
Install
Python 0.3.17 211K
Linting, Debugging (multi-threaded, remo...
Don Jayamanne
Install
Debugger for Chrome 148K
Debug your JavaScript code in the Chro...
Microsoft JS Diagnostics
Install
C/C++ 0.7.1 143K
Complete C/C++ language support inclu...
Microsoft
Install
Go 0.6.39 99K
Rich Go language support for Visual Stu...
lukehoban
Install
ESLint 0.10.18 88K
Integrates ESLint into VS Code.
Dirk Baeumer
Install
PowerShell 0.6.1 85K
Develop PowerShell scripts in Visual Stu...
Microsoft
Install
Material-theme 0.0.1 66K
Material themes for VS Code. By ZTF
zhuangtongfa
Install
app.ts
www.ts
package.json
README.md
import app from './app';
import debugModule = require('debug');
import http = require('http');
const debug = debugModule('node-express-typescript:ser
// Get port from environment and store in Express.
const port = normalizePort(process.env.PORT || '3000')
app.set('port', port);
CSSImportRule
CSSSupportsRule
export
exports
import
importScripts
MessagePort
normalizePort
port const port: number | string | boolean
*/
function normalizePort(val: any): number|string|boolea
let port = parseInt(val, 10);
if (isNaN(port)) {
// named pipe
return val;
}
if (port >= 0) {
master 1↓ 13↑ 0 0
Ln 9, Col 21 Spaces: 2 UTF-8 LF TypeScript

Electron

- Not a C++ library
- **A WebView for JavaScript**
- node.js for GUI
    - Same API
    - Same Native modules
- Everything (the BRIB) is node.js-like JavaScript
- **A JavaScript programmer's dream come true**
- It can still work for you

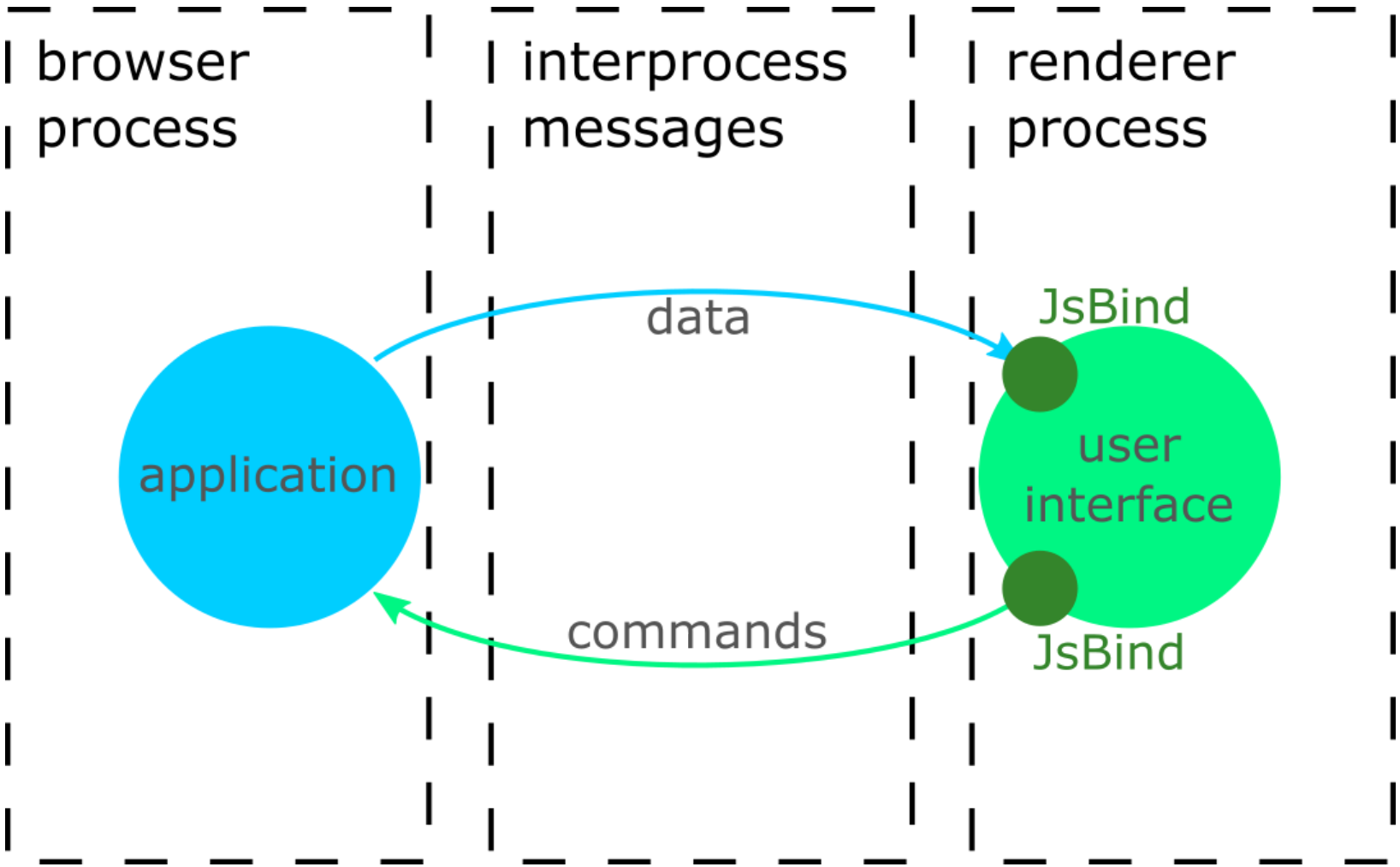
Electron Modules



Qt WebEngine and CEF

- Good Chromium wrappers
- Almost the exact same features
- The main difference is the license
- More good Words about CEF
  - C interface
  - Java, .NET, Go, Python...
  - Electron used to be CEF
- **CEF Echo demo**
- **CEF File system browser demo**
- JsBind - **gh/Chobolabs/jsbind**

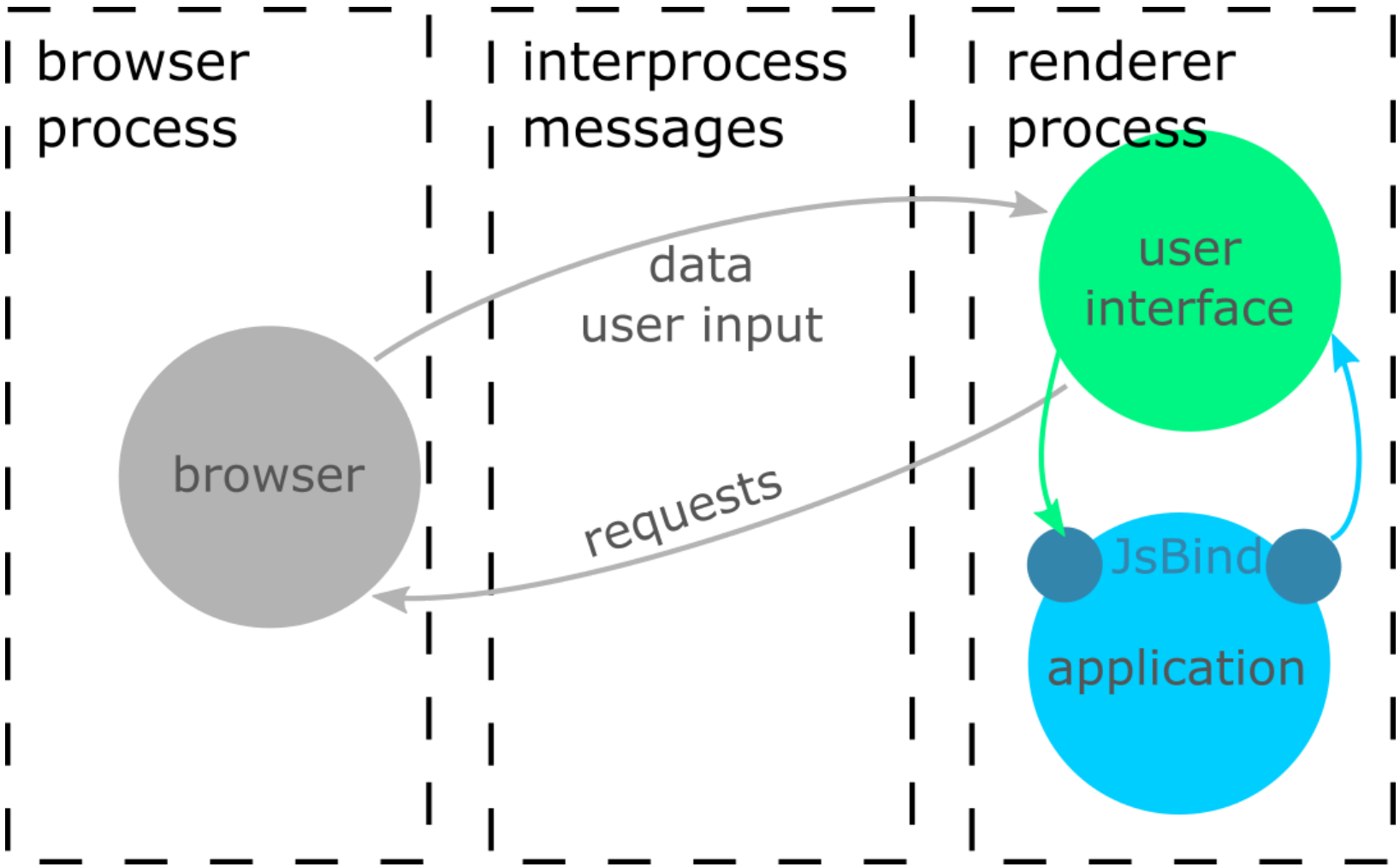
CEF and JsBind



Yet more variants

- Putting it inside existing UI. Qt WebEngine is the winner
- Custom rendering
    - Everything must be in the browser. Or...
    - Render offscreen. Send to browser
    - Get the browser screen. **Demo**
- Lots of load on interprocess communication
    - Everything in the renderer process
    - Don't forget to turn off sandboxing
    - **Synchronous CEF demo**
    - Share memory

Renderer Process Application



Browser embedding pros

- Controller code in C++
- Combine browser GUI with other GUI
- Ability to work entirely in the renderer process

Browser embedding cons

- The **fancy workflow is lost**

Browser embedding cons

- ~~The **fancy workflow is lost**~~
  - We are embedding a browser which can open localhost
  - **Remote Dev Tools demo**
  - Mocking is also your friend. FS-Browser uses mocking
- **Debugging multiple processes** is harder
  - Tooling to the rescue
  - Single-process mode. Use with caution
- We want to go faster
  - Custom rendering
  - Synchronous coms and sharing memory
  - Sharing video memory

Approach 3

Custom HTML renderers

Typically game-dev territory

Take the rendering process out of the browser

Characteristics

- Similar to many GUI libraries
- Graphics only
- Bring Your Own Renderer
- At least some DOM and API changes
- `<img src=id123 />`

(Almost) Complete Solutions

- No known FOSS solutions
- **Ultralight** - **ultralig.ht**
  - Free for free software
  - Stripped down WebKit
- **Coherent Gameface** - **coherent-labs.com**
  - Not free
  - Complete custom implementation
  - **CppCon 2018: Stoyan Nikolov OOP Is Dead, Long Live Data-oriented Design**

Partial Solutions

- **Qt Quick/QML** - Qt. Custom HTML/CSS Custom DOM
- **sciter** - Quite poweful. No JS. Free without source and support
- **libRocket** - FOSS. Dead: lives in forks. Custom DOM. No JS.
- **RmlUi** - FOSS. Custom HTML/CSS. Custom DOM. No JS.
- **litehtml** - FOSS. No DOM. No JS.

Approach 4

C++ ⇒ WebAssembly or asm.js

Just for completeness

Join me **in Crest 3 at 20:30** for more

End

Questions?

Addendum session in Crest 3 at 20:30

Borislav Stanimirov / **ibob.github.io** / **@stanimirovb**

Demos: **github.com/iboB/html5-gui-demo**

These slides: **ibob.github.io/slides/html5-gui/**