

Headless Chrome

skyostil@ / alexclarke@
BlinkOn 6

```
houses the documentation and code related to the  
Chromium projects and is intended for developers  
interested in learning about and contributing to the  
open-source projects.<br />  
</div>  
</div>  
  
<table cellpadding="0"  
  class="sites-layout-hbox">  
  <tbody>  
    <tr>  
      <td class=  
        "sites-layout-tile sites-tile-name-content-1">  
        <div dir="ltr">  
          <b><a href=  
            "http://www.chromium.org/Home"><span style=  
              "font-size:large">Chromium</span></a> </b>  
          Chromium is an open-source browser proje  
          aims to build a safer, faster, and mor  
          way for all users to experience the  
          site contains design documents, an  
          overviews, testing information,  
          you learn to build and work wi  
          source code.  
        </div>  
      </td>  
    </tr>  
  </tbody>  
</table>
```

Headless mode

- Browser without a UI
- Able to run in a server environment
- Intended for testing and automation

Out of scope:

- Not an embeddable browser widget

Motivation

- Existing solutions
 - Regular browser using virtual framebuffer (xvfb)
 - PhantomJS, NW.js (node-webkit)
 - Frameworks such as Chromium Embedded Framework, QtWebEngine, etc.
- Clients
 - Selenium & WebDriver
 - Karma test runner
 - Chromium Telemetry, layout test runner

Motivation

- Problems
 - High resource usage
 - Painful to set up
 - No support for modern web features (e.g., ES6, Service Workers, WebGL, ...)
 - No auto-updates
 - Non-deterministic behavior
 - May have less security isolation



Sam Saccone

@samccone



Follow

Headless Chrome is coming so soon 🎪!!! Say
goodbye to the myriad of hacks to run Chrome
and phantom on CI. 🌐

[bugs.chromium.org/p/chromium/iss ...](https://bugs.chromium.org/p/chromium/iss...)

RETWEETS

1,706

LIKES

1,944



11:47 am - 4 Jun 2016



Mountain View, CA



1.7K



1.9K






Sam Saccone

@samccone



Follow

Headless Chrome is coming so soon 🎪!!! Say
goodbye to the myriad of hacks to run Chrome
and phantom on CI. 

[bugs.chromium.org/p/chromium/iss ...](https://bugs.chromium.org/p/chromium/iss...)

RETWEETS

1,706

LIKES

1,944



11:47 am - 4 Jun 2016

📍 Mountain View, CA



1.7K



1.9K

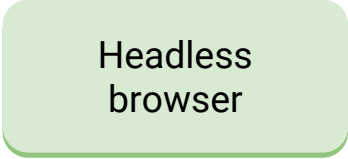


Issue 546953

Starred by 169 users

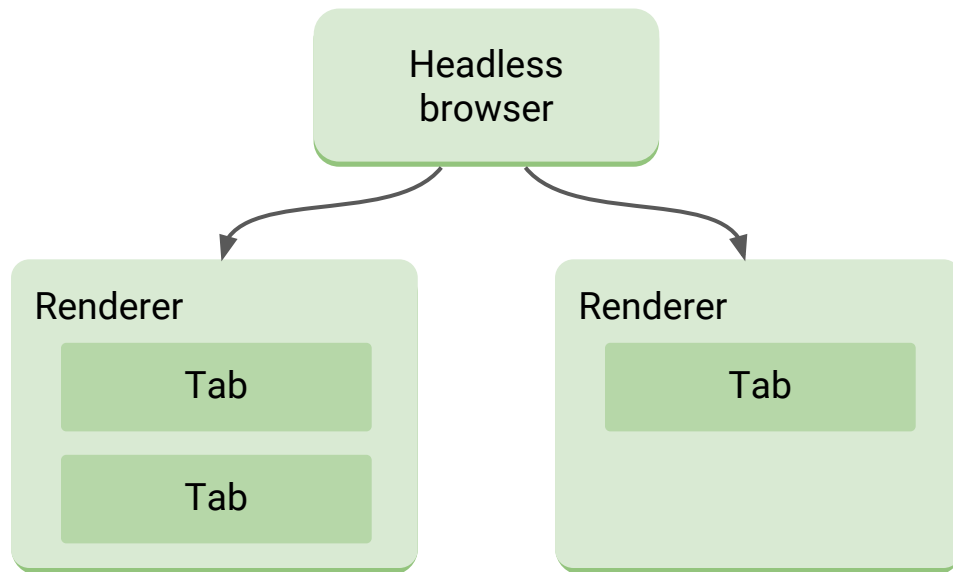
Headless Chrome

Headless Chrome

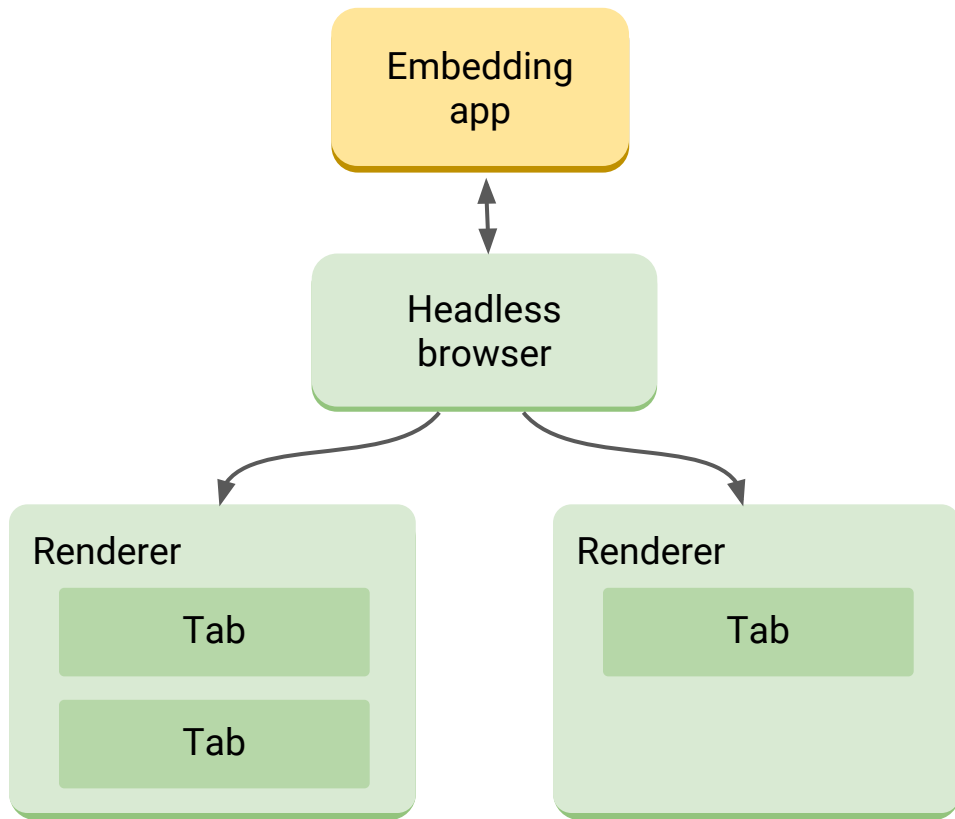


Headless
browser

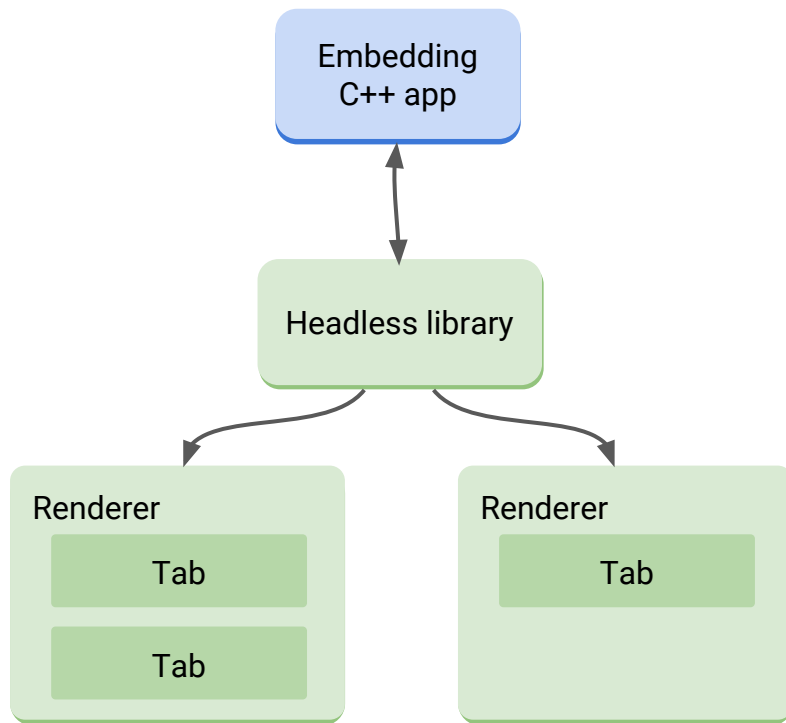
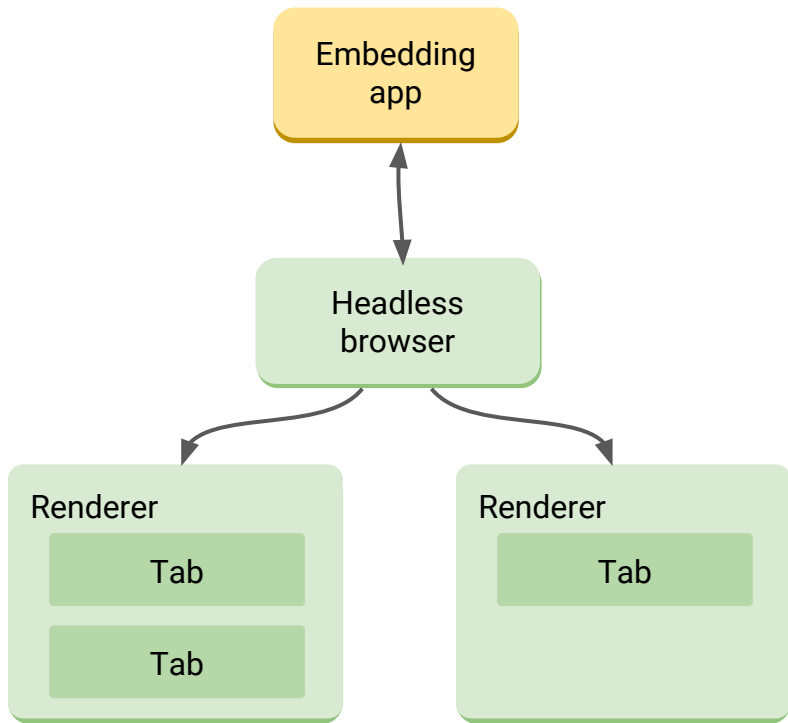
Headless Chrome



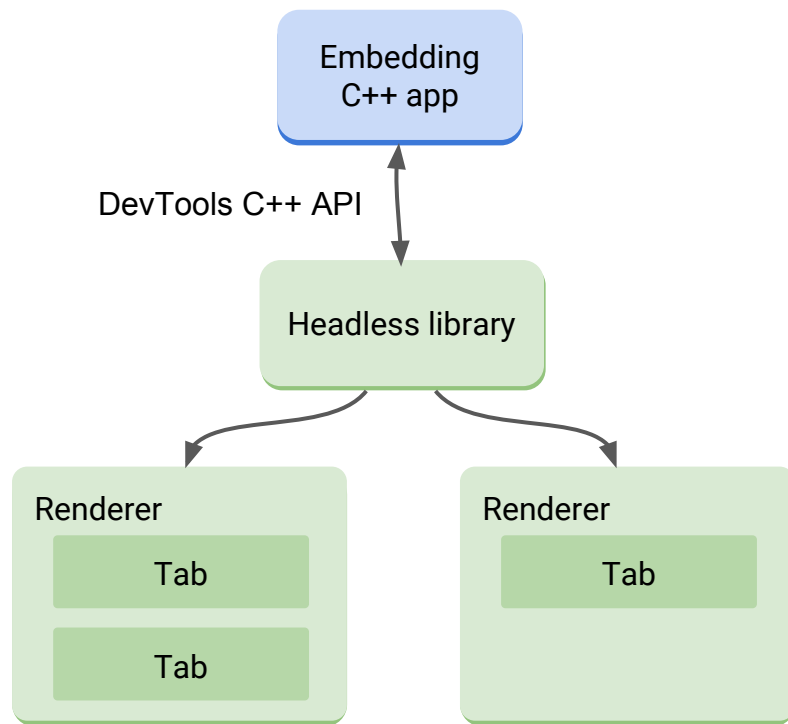
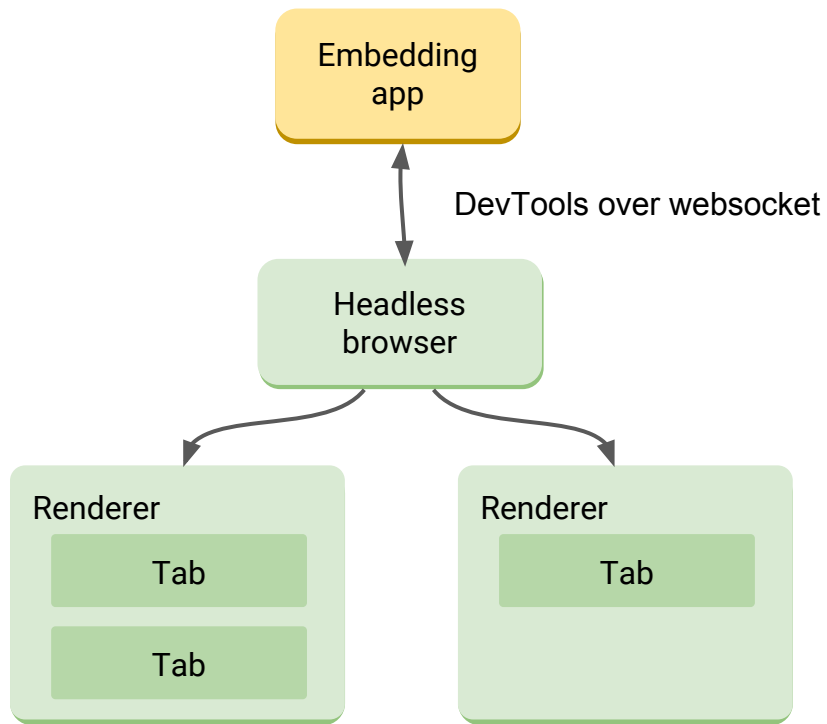
Headless Chrome



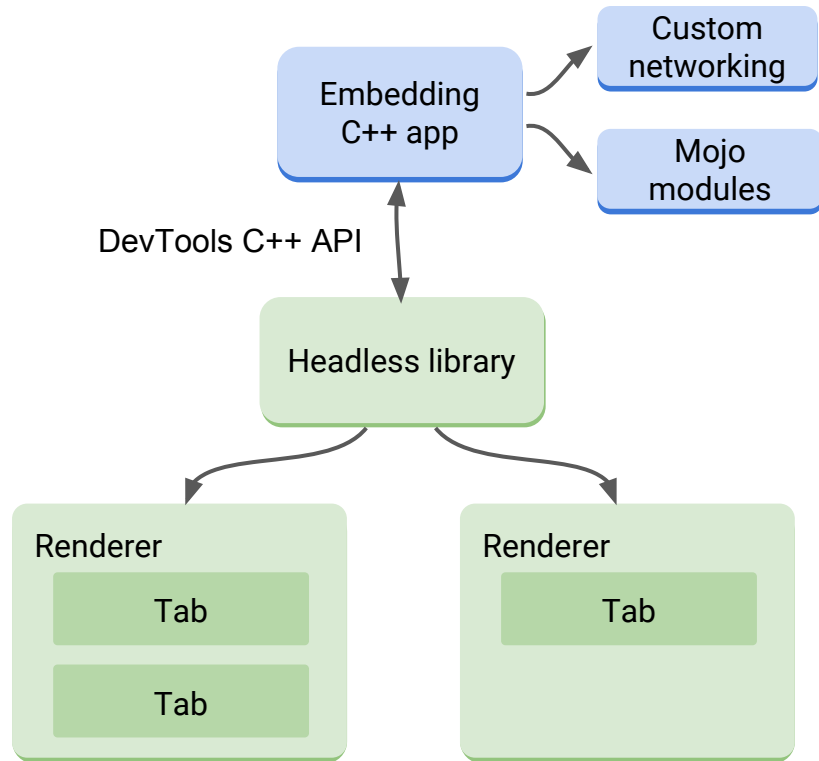
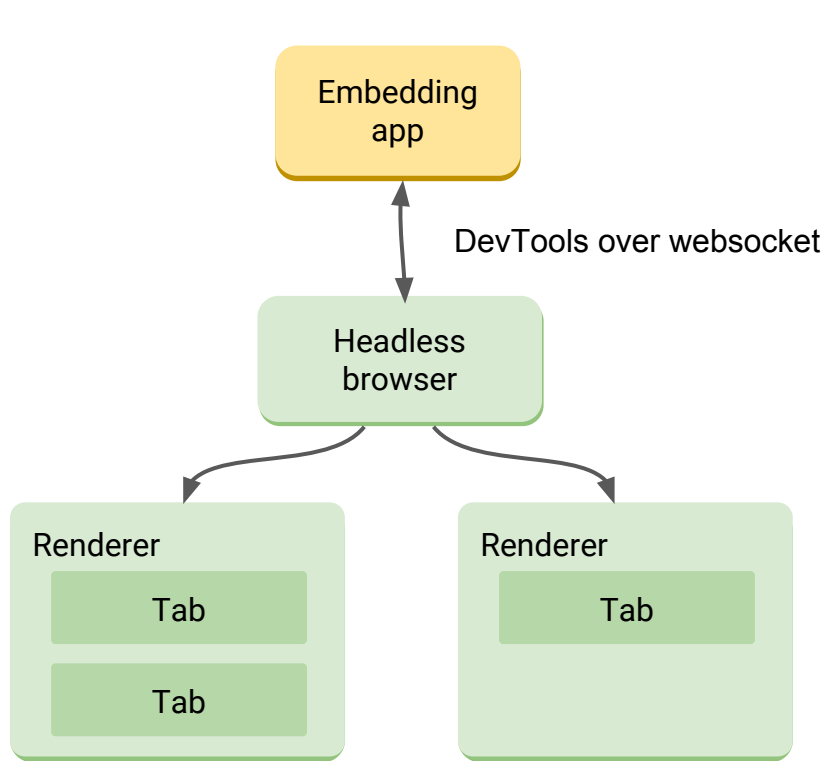
Headless Chrome



Headless Chrome

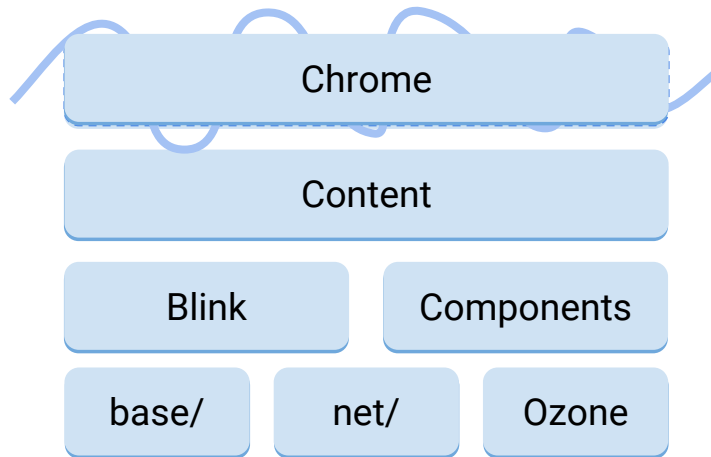


Headless Chrome



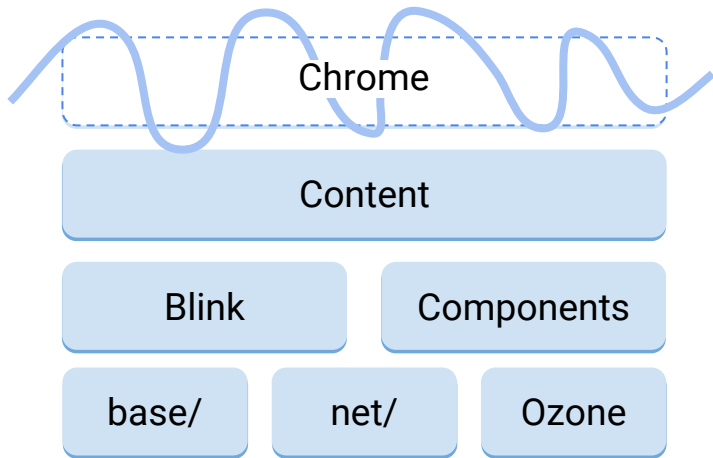
Headless Chrome

- Differences from regular Chrome
 - No Chrome-level features (e.g., profiles, sync, safe browsing, ...)
 - No UI – offscreen software rendering
 - No extensions (for now)
 - No audio



Headless Chrome

- Differences from regular Chrome
 - No Chrome-level features (e.g., profiles, sync, safe browsing, ...)
 - No UI – offscreen software rendering
 - No extensions (for now)
 - No audio
- What you do get:
 - Everything below the content layer
 - Blink with the latest web platform awesomeness
 - Profiling and introspection tools (DevTools UI, tracing, etc.)



Headless additions

- More DevTools API capabilities for everyone
 - Web data extraction
 - DevTools forwarding
 - Opening and closing tabs
- Determinism
 - Session isolation
 - Virtual time
 - Deterministic time, date, random numbers, etc.
- DevTools bindings for C++ and Javascript
- Network customization (C++)
- Mojo modules (C++)

- More flexible screenshots
 - Layout & visual viewport controls
- DOM dumping
 - Atomic snapshotting of the DOM for the entire page
- Chrome traces
- Resource fetches
- More data about the layout tree

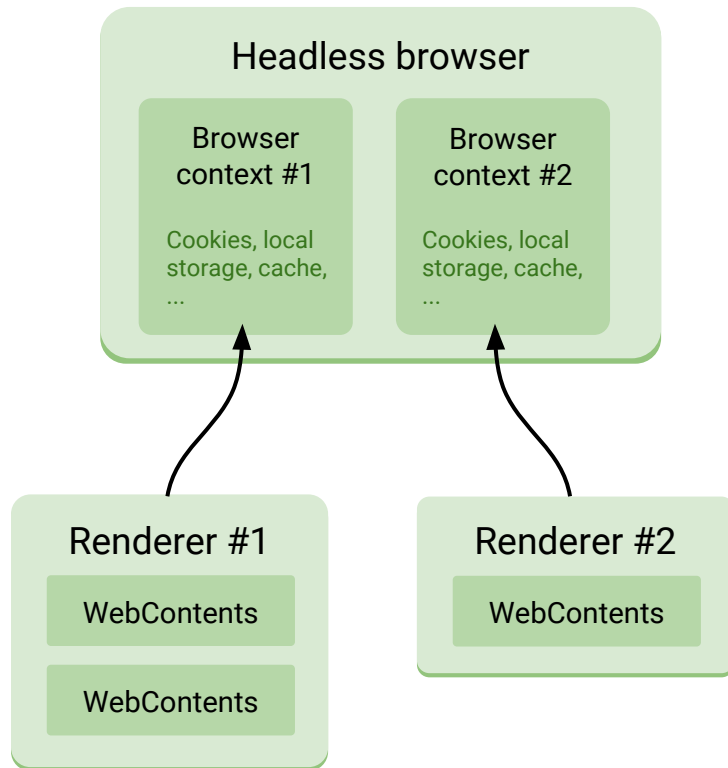
[illegible]

Determinism: Session isolation

- Save resources by running parallel sessions in a single browser
- Each session gets unique
 - Cache
 - Cookies
 - Local storage
 - ...
- Session serialization
 - Avoids need to restart browser or mess with file system to get a clean state

Browser process

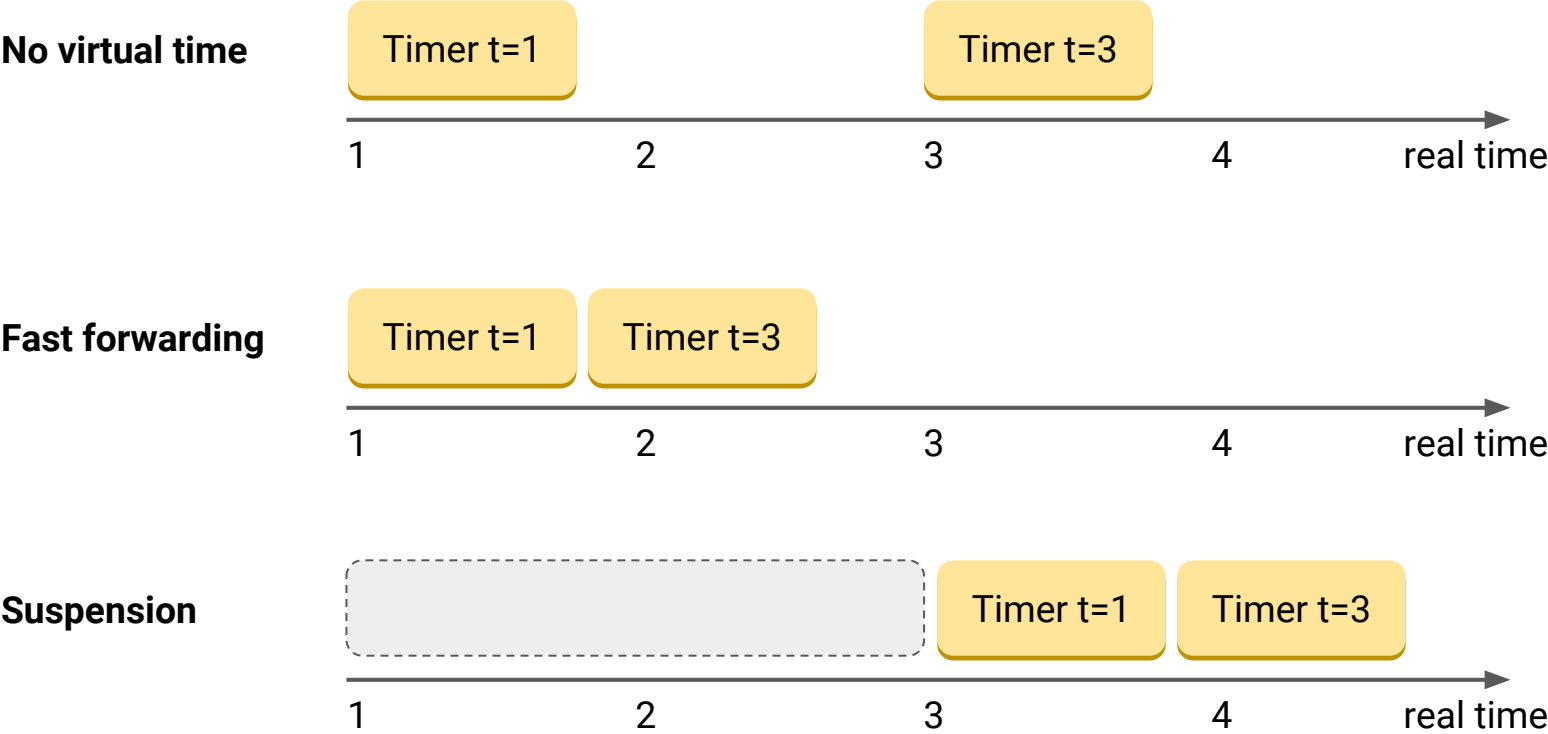
Renderer processes



Determinism: Virtual time

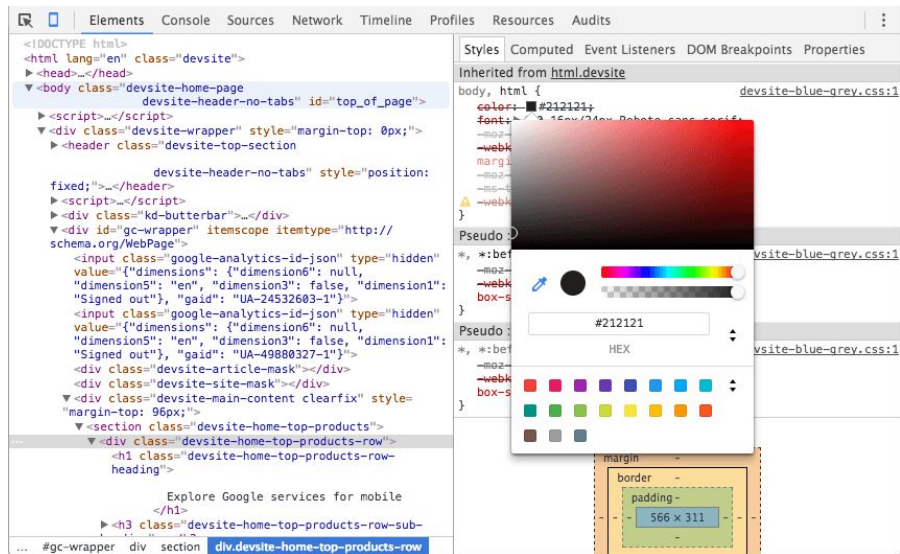
- Timers can use virtual time instead of real time
- Examples
 - Fast-forward time to speed up page loading
 - Stop time while network requests are pending
 - Suspend and resume renderer

Determinism: Virtual time

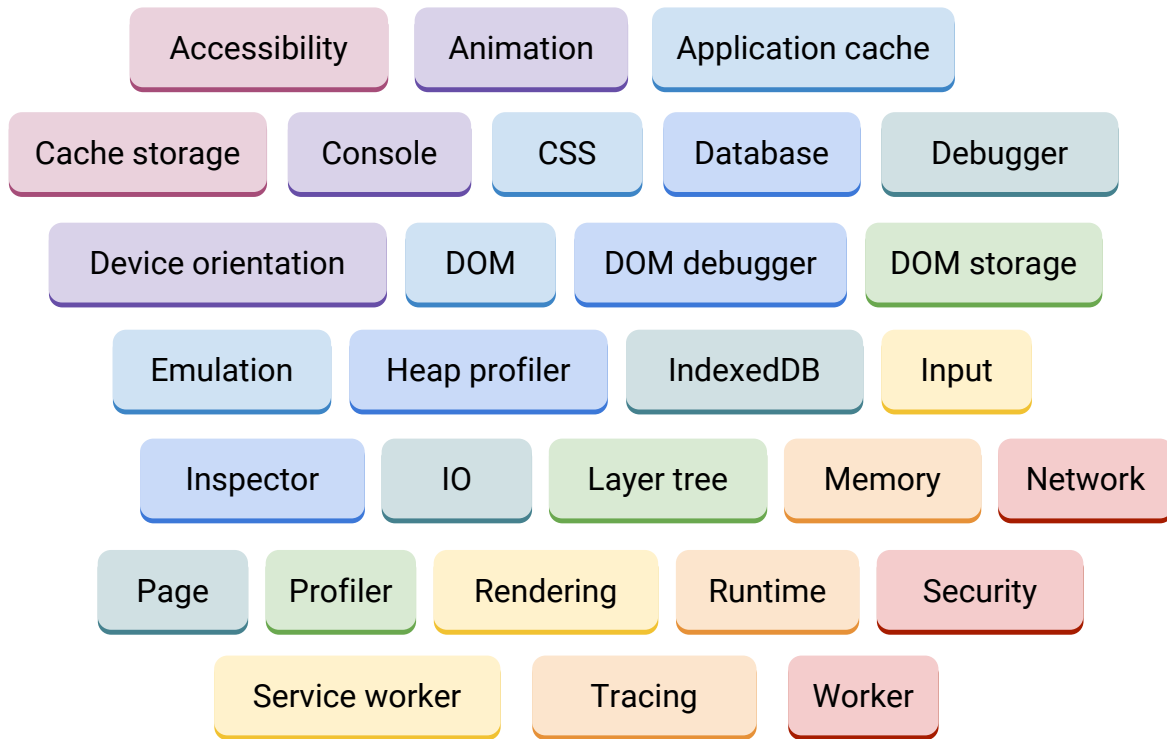


Controlling the browser: DevTools protocol

- Magic behind Chrome Developer Tools
- Allows controlling, inspecting and debugging pages
- Commands, events



Controlling the browser: DevTools protocol



DevTools wire protocol



```
graph LR; A[DevTools client] --- B[DevTools target];
```

DevTools
client

DevTools target

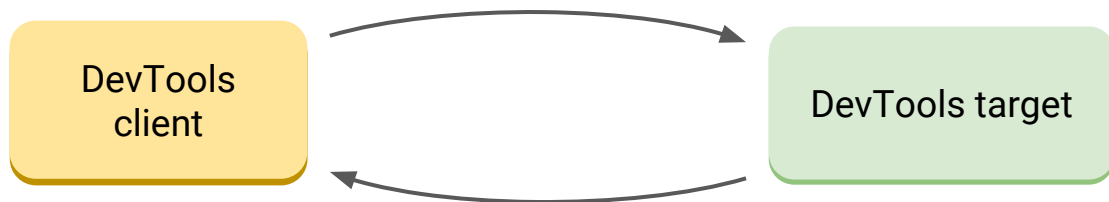
DevTools wire protocol

```
{id: 1, method: "Runtime.evaluate", params: {  
  expression: "1 + 1"  
}}
```



DevTools wire protocol

```
{id: 1, method: "Runtime.evaluate", params: {  
  expression: "1 + 1"  
}}
```



```
{id:1, result: {  
  result: {  
    type: "number", value: 2, description: "2"  
  },  
  wasThrown: false  
}}
```

DevTools wire protocol



```
{method: "Page.loadEventFired", params: {  
  timestamp: 376371.511946  
}}
```

DevTools bindings for Headless

- Two flavors
 - C++
 - Javascript
- Type-safe
- Generated for a specific protocol version
 - Backwards compatible
- Stable and experimental APIs

DevTools bindings for Headless

C++

```
client->GetRuntime()->Evaluate("1 + 1", base::Bind(&OnResult));

void OnResult(std::unique_ptr<headless::runtime::EvaluateResult> result) {
    std::string value;
    result->GetResult()->GetValue()->GetAsString(&value);
    std::cout << value << std::endl;
}
```

DevTools bindings for Headless

```
C++ client->GetRuntime()->Evaluate("1 + 1", base::Bind(&OnResult));

void OnResult(std::unique_ptr<headless::runtime::EvaluateResult> result) {
    std::string value;
    result->GetResult()->GetValue()->GetAsString(&value);
    std::cout << value << std::endl;
}
```

```
JS client.runtime.evaluate("1 + 1").then(response => {
    console.log(response.result.value);
});
```

DevTools bindings for Headless

```
C++ client->GetRuntime()->Evaluate("1 + 1", base::Bind(&OnResult));

void OnResult(std::unique_ptr<headless::runtime::EvaluateResult> result) {
    std::string value;
    result->GetResult()->GetValue()->GetAsString(&value);
    std::cout << value << std::endl;
}
```

```
ES6 let response = await client.runtime.evaluate("1 + 1");
    console.log(response.result.value);
```

DevTools bindings for Headless

C++

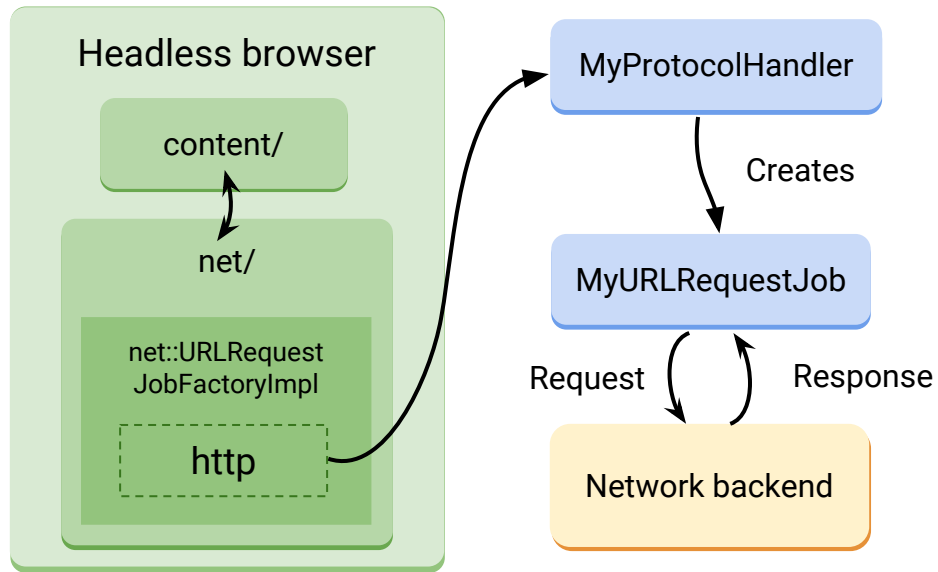
```
client->GetPage()->Enable();  
client->GetPage()->AddObserver(this);  
  
void MyObserver::OnLoadEventFired(  
    std::unique_ptr<page::LoadEventFiredParams> params) override {  
    std::cout << params->GetTimestamp() << std::endl;  
}
```

JS

```
client.page.enable();  
client.page.addLoadEventFiredListener(params => {  
    console.log(params.timestamp);  
});
```

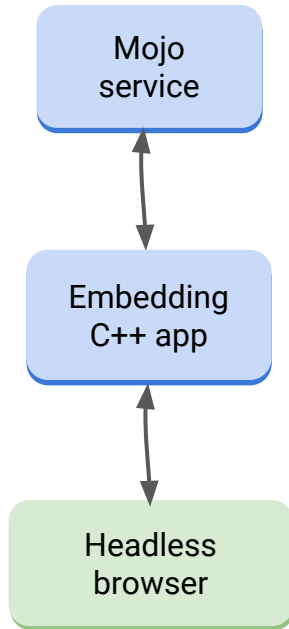

Network customization (C++)

- Implement custom handlers for `http://`, `https://`, `file://`, ...
- Examples
 - Fetch pages from disk instead of the internet
 - Replace specific resources
 - Runtime generated response (e.g., for a test harness)



Mojo services

- Provide JS bindings to C++ objects
- Examples
 - Test runner objects (e.g., `testRunner.dumpAsText()`)
 - Host OS interfaces (e.g., enabling airplane mode)



Mojo services



```
graph LR; MS[Mojo service] --- R[Renderer]
```

Mojo
service

Renderer

Mojo services

.mojom

```
module example;  
MyService {  
  SendTestResults(string result);  
};
```

Mojo
service

Renderer

Mojo services

.mojom

```
module example;  
MyService {  
  SendTestResults(string result);  
};
```

JS

```
Mojo.example.MyService.then(service => {  
  service.sendTestResults("all passed");  
});
```



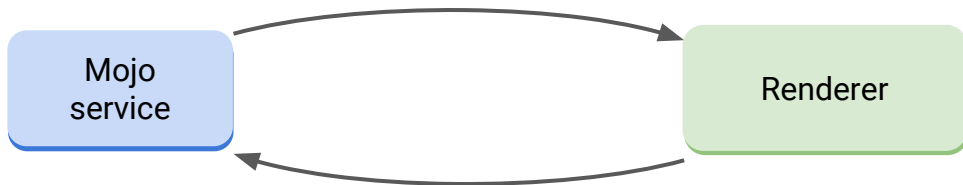
Mojo services

.mojom

```
module example;  
MyService {  
  SendTestResults(string result);  
};
```

JS

```
Mojo.example.MyService.then(service => {  
  service.sendTestResults("all passed");  
});
```



C++

```
class MyServiceImpl : public example::MyService {  
  void SendTestResult(const mojo::String& result) override {  
    EXPECT_EQ("all passed", result.get());  
  }  
};
```

Distribution

- Chrome in headless mode

```
$ chrome --headless --remote-debugging-port=9222
```

- Doesn't quite exist yet
- Experimental version: Headless Shell

- C++ library

- Lives in headless/ in the Chromium repository
- Distributed as source code (no stable ABI)
- See headless/README.md (bit.ly/23LR2pW)

Future work

- Finish all the things we promised in these slides :)
- Test harness integration (WebDriver, Selenium)
- Windows & ~~OSX~~ macOS support
- Node.js package
- Reduce memory footprint
- WebGL (software rendering and/or offscreen GPU)
- Chrome extensions (maybe)
- Trace analysis for guiding performance optimization

Give it a try

```
$ mkdir -p out/Debug
```

```
$ echo 'import("//build/args/headless.gn")' > out/Debug/args.gn
```

```
$ gn gen out/Debug
```

```
$ ninja -C out/Debug headless_shell
```

```
$ out/Debug/headless_shell https://www.chromium.org
```

Mailing list: headless-dev@chromium.org (bit.ly/1ZBcyLx)

Documentation: bit.ly/23LR2pW Bug label: **Proj-Headless**