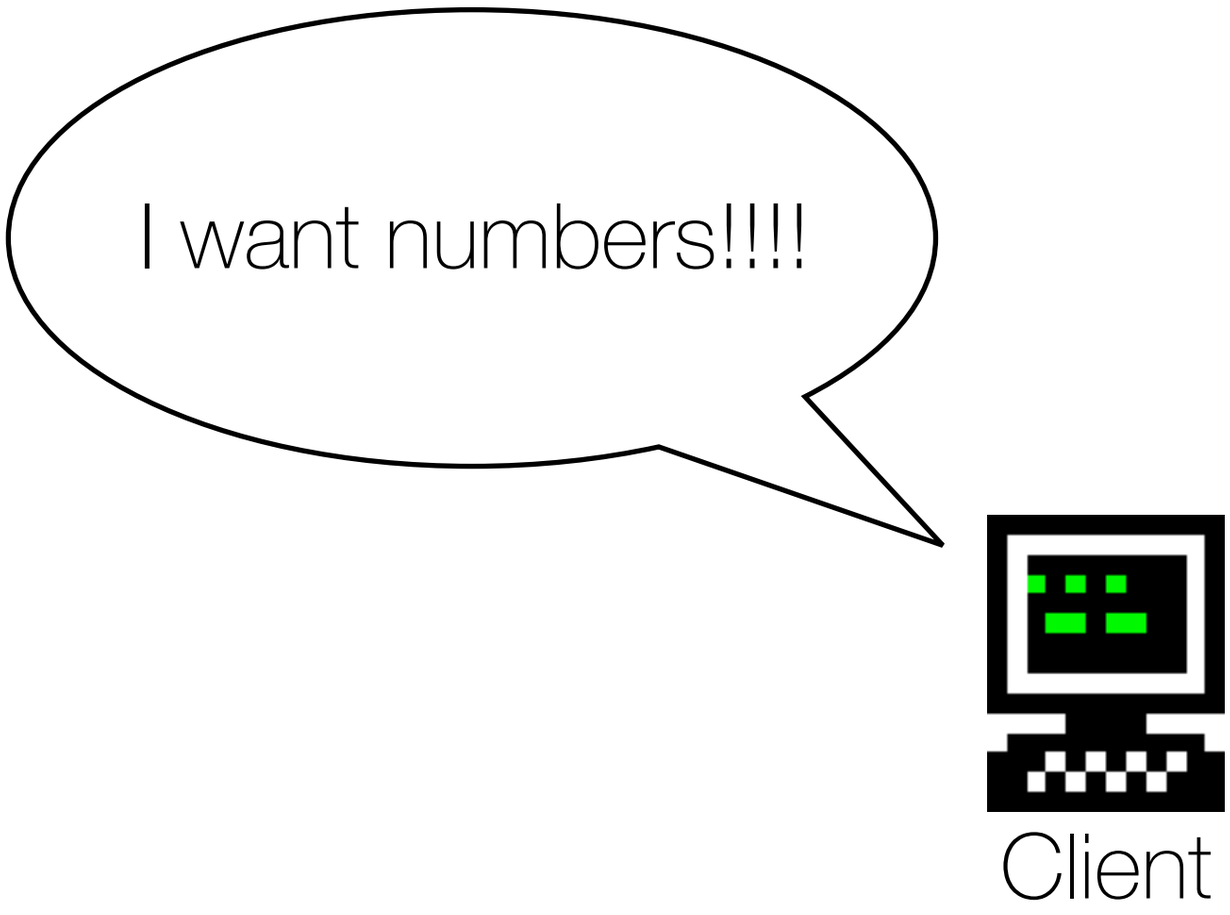
Server Farm to Table

Or, How the Internet Works.

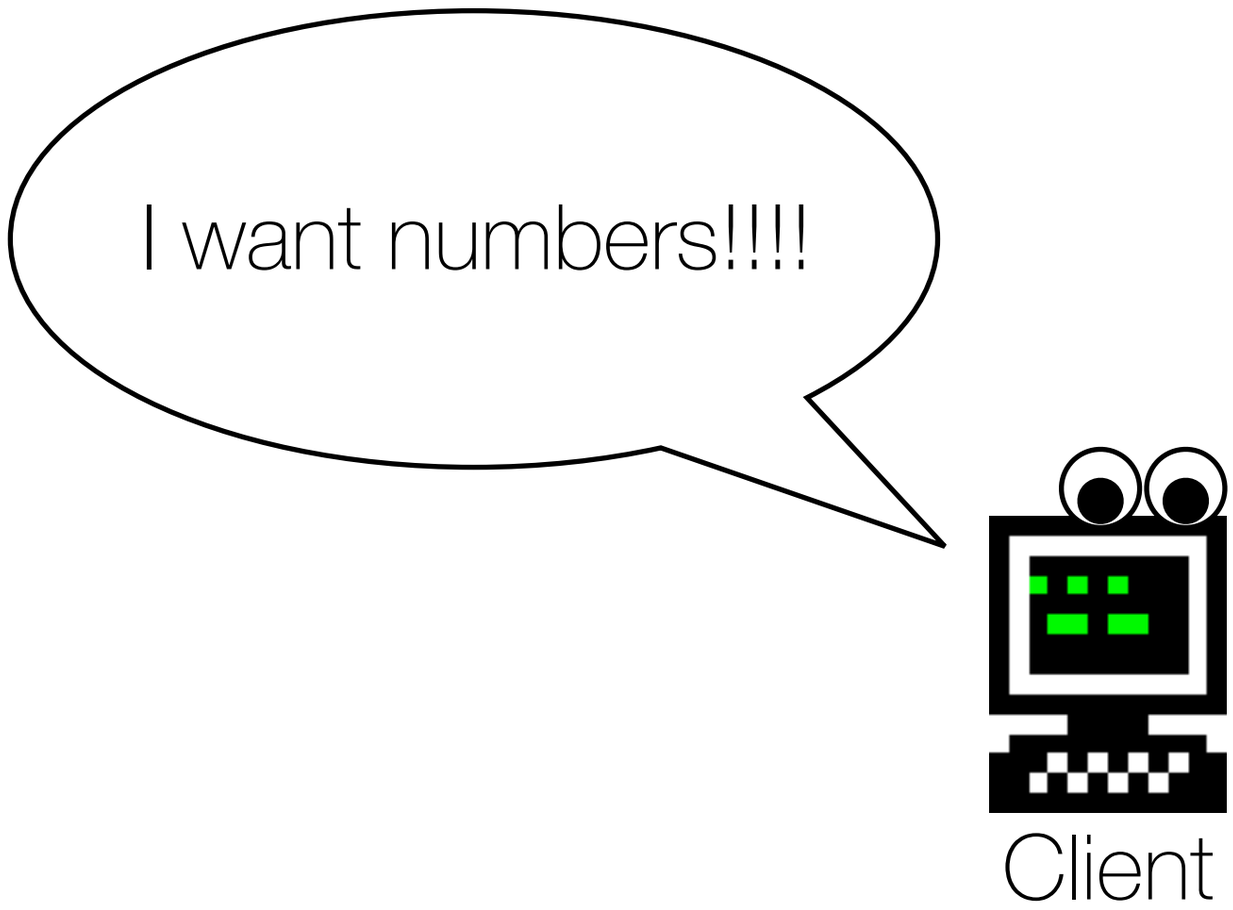
0. User Enters URL

0. User Enters URL

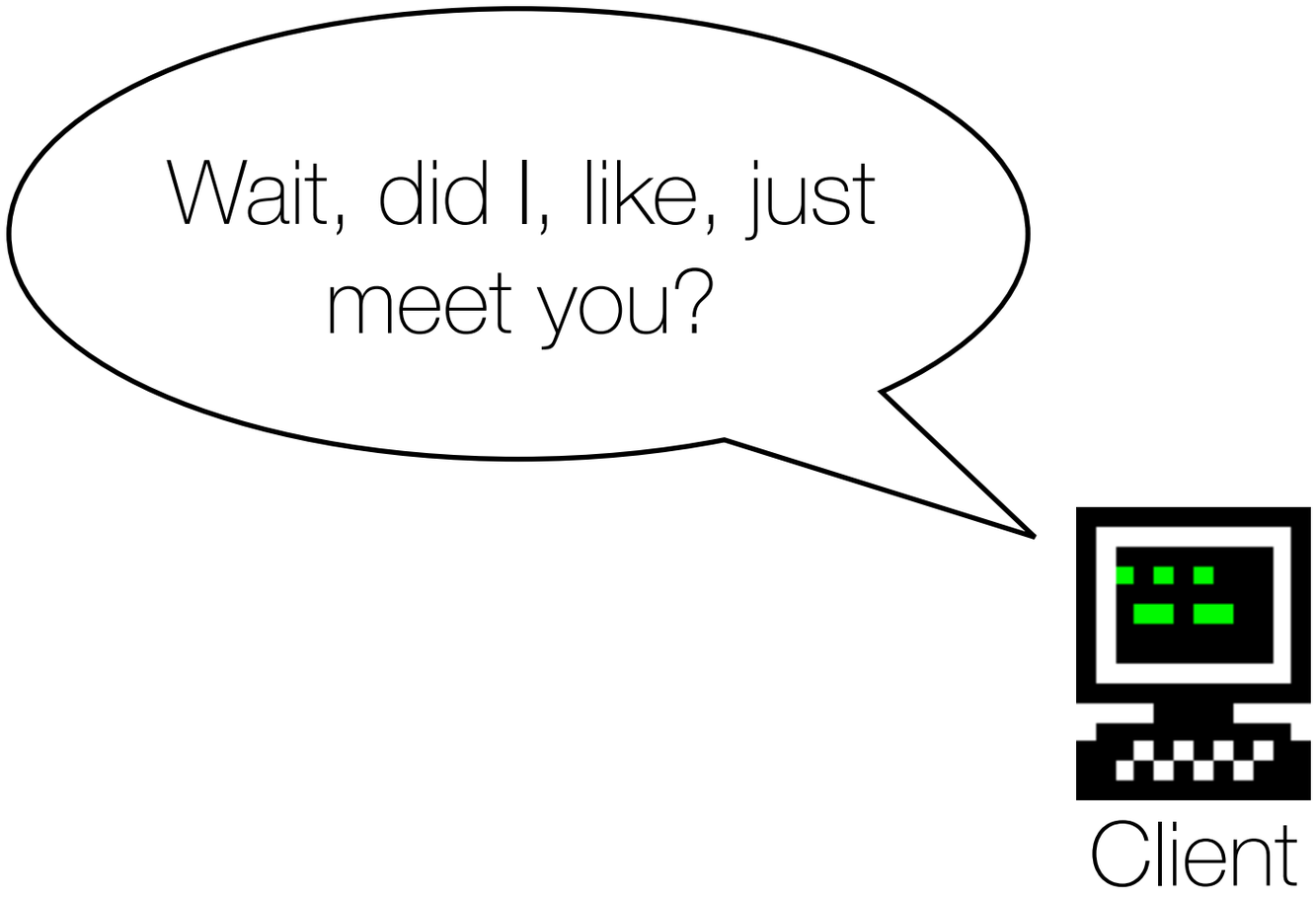
1. IP Address Lookup



1. IP Address Lookup



1. IP Address Lookup



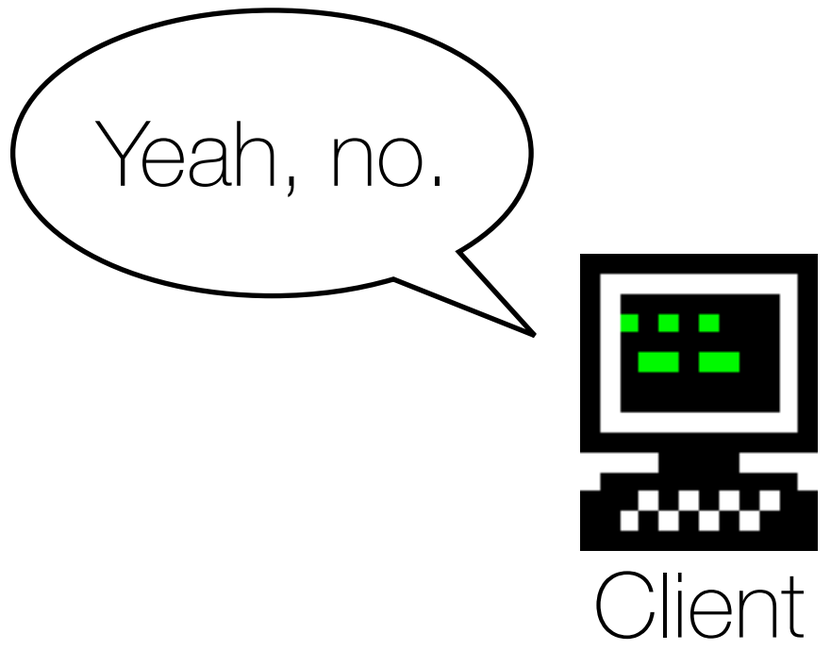
Browser checks its cache.

1. IP Address Lookup

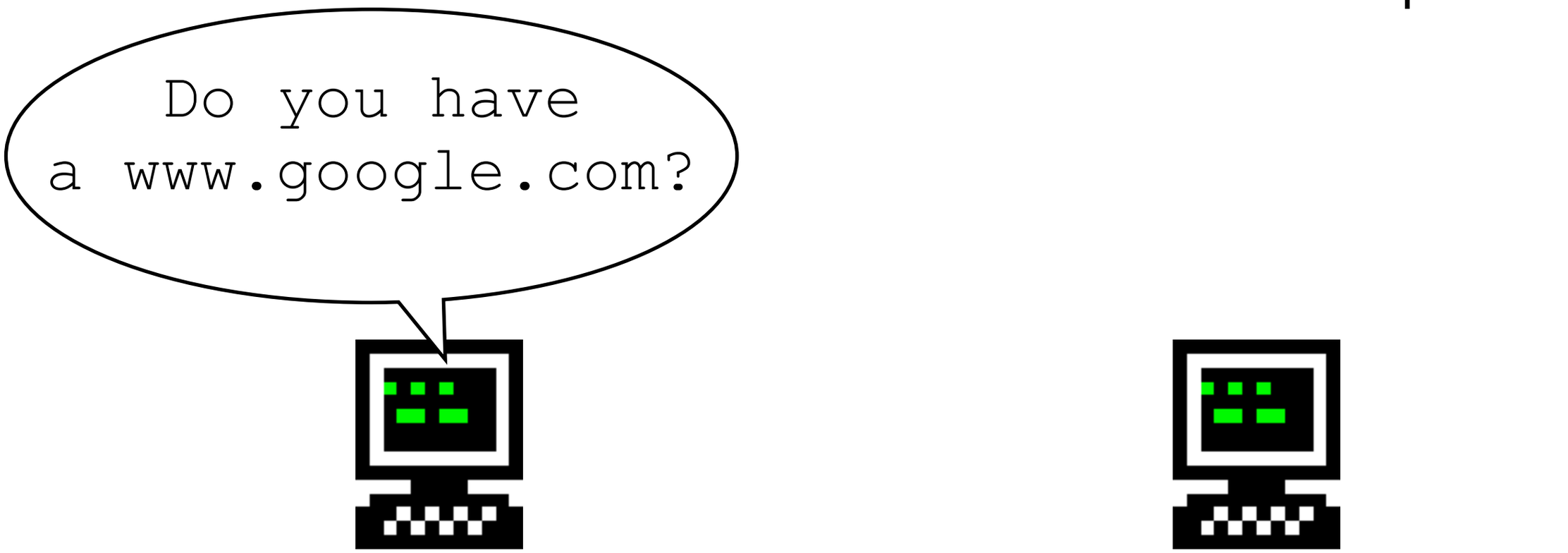
Client

Browser checks /etc/hosts

1. IP Address Lookup

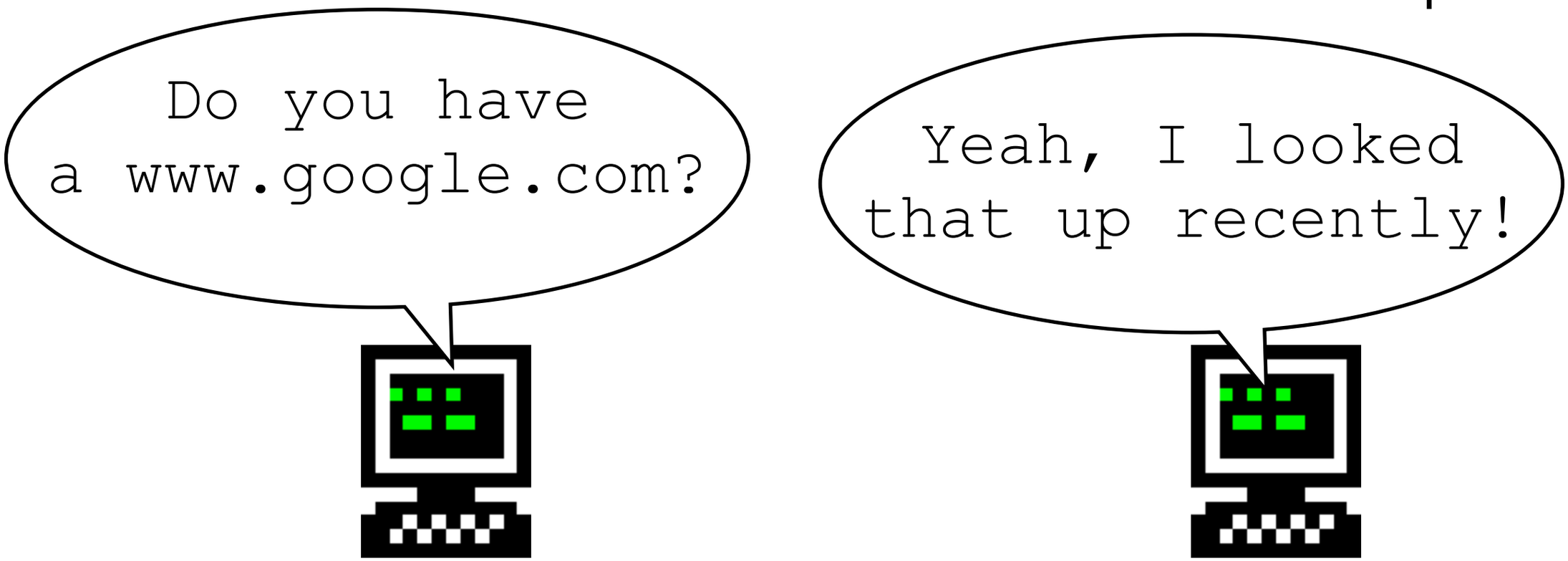


1. IP Address Lookup



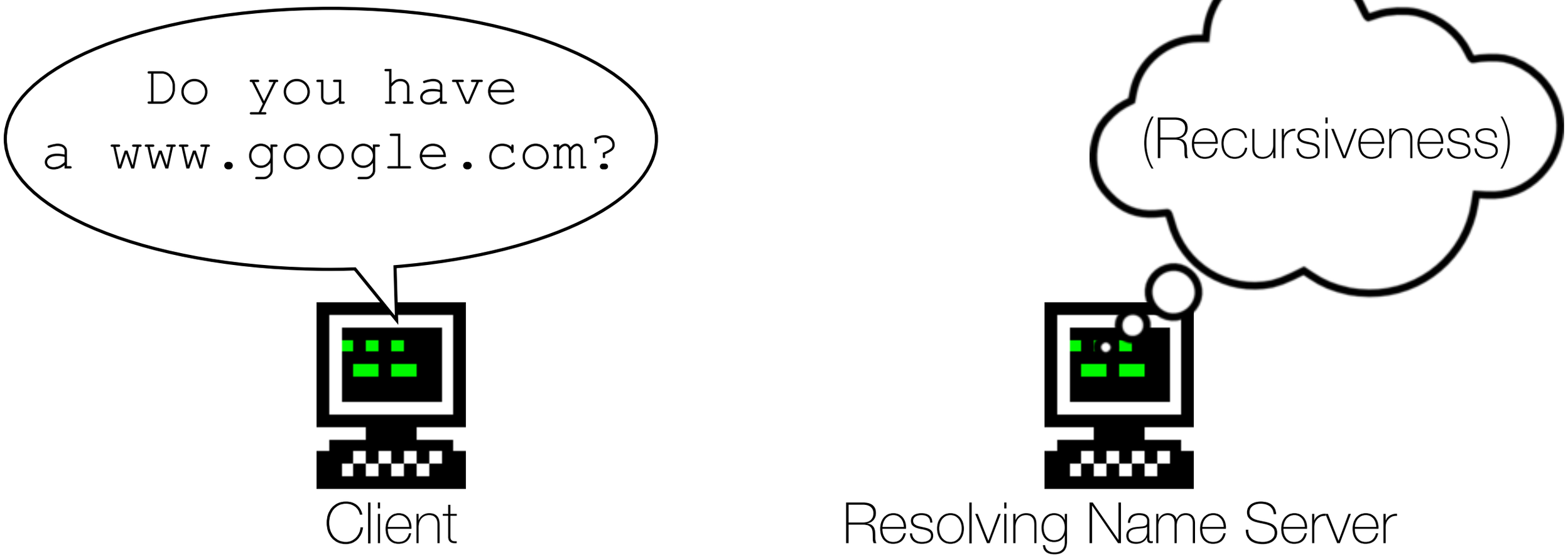
Browser hits DNS server which…

1. IP Address Lookup



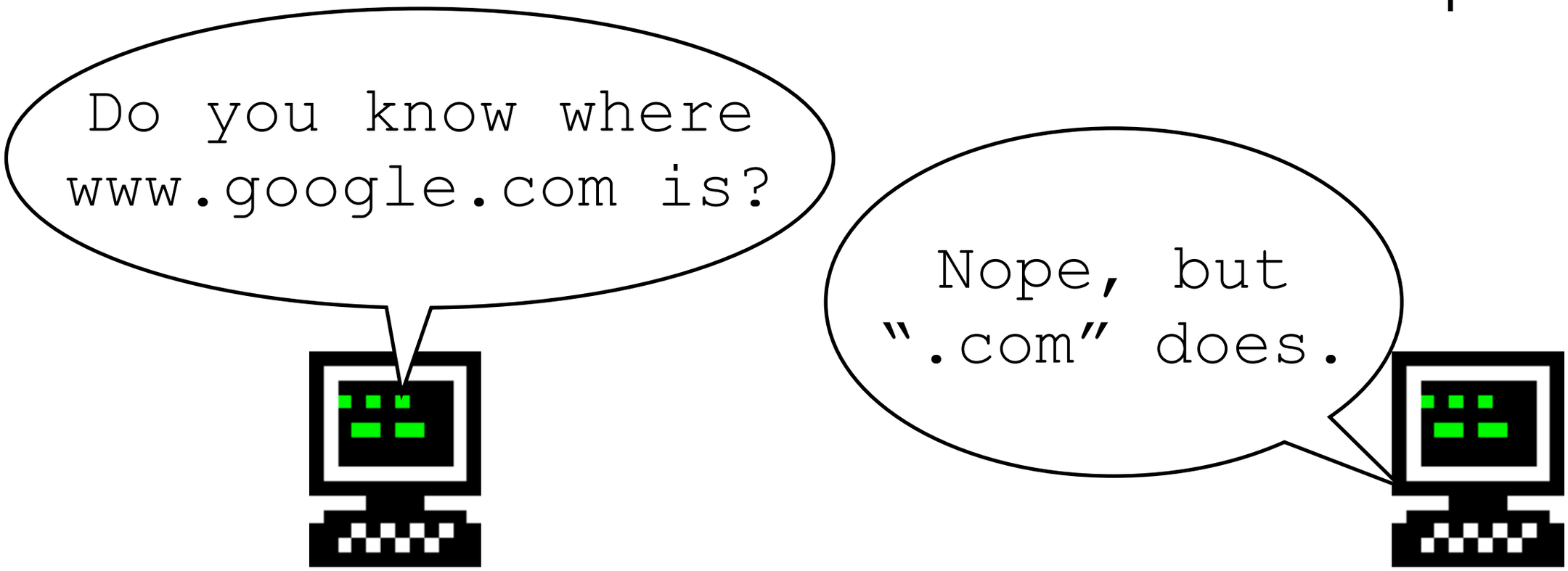
…first looks in it's cache…

1. IP Address Lookup



… or it asks a bunch more servers

1. IP Address Lookup

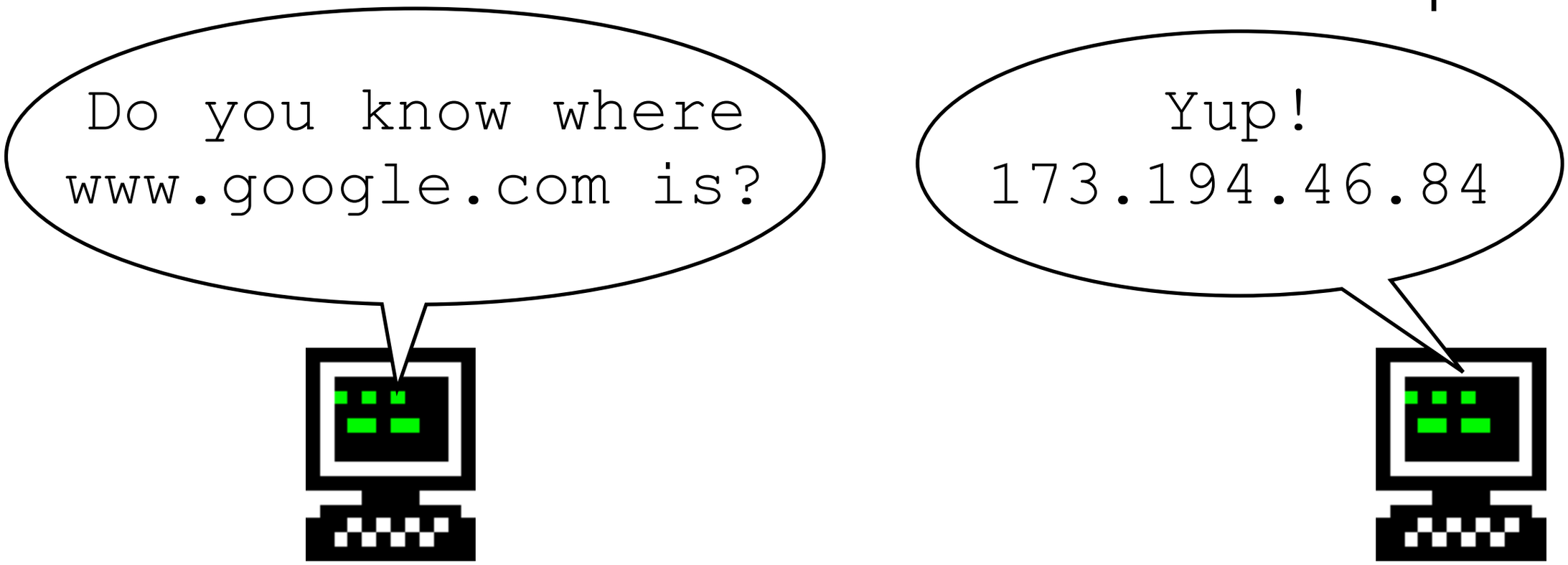


Browser hits DNS server which returns answer
(…after it asks a bunch more servers)

1. IP Address Lookup

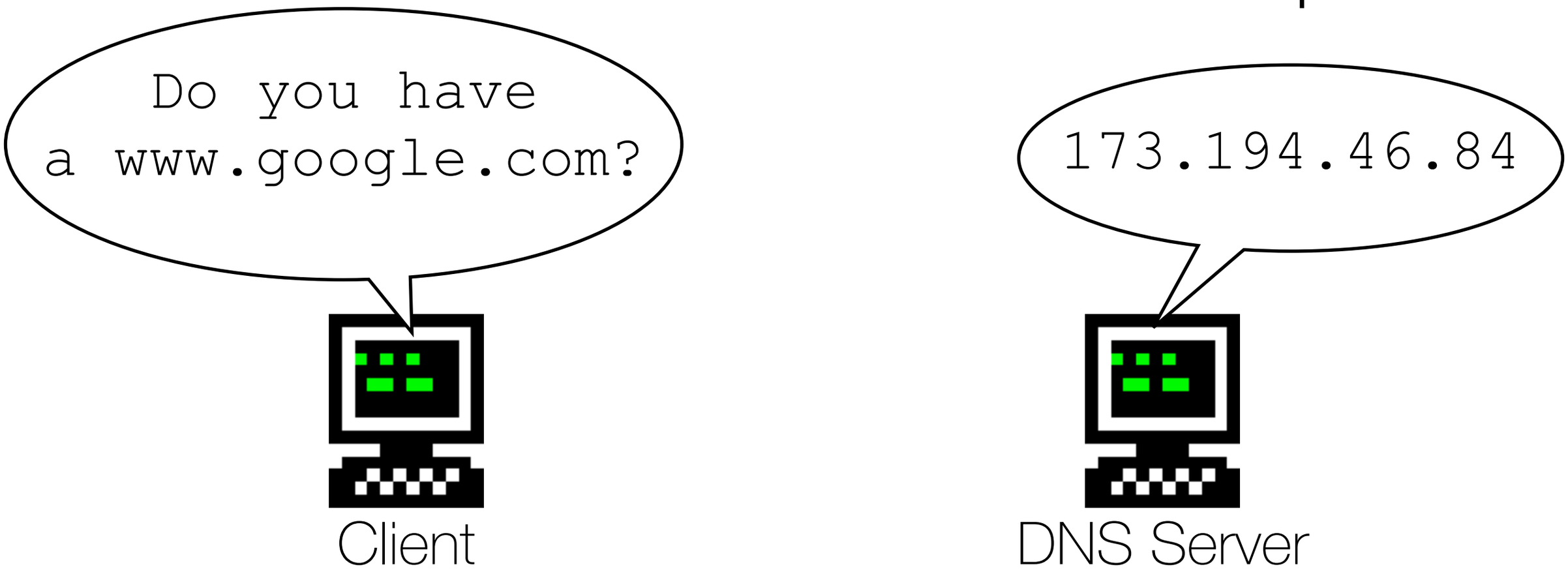
Browser hits DNS server which returns answer
(…after it asks a bunch more servers)

1. IP Address Lookup



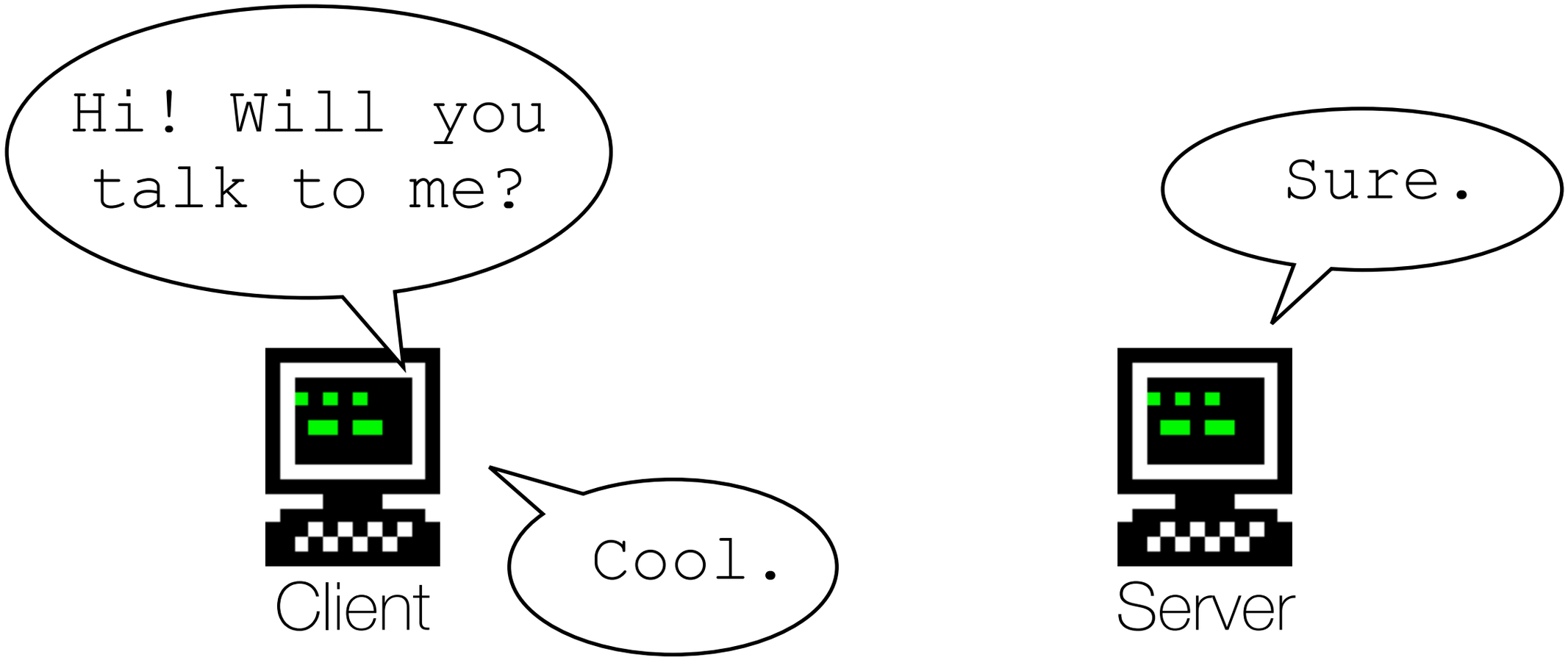
Browser hits DNS server which returns answer
(…after it asks a bunch more servers)

1. IP Address Lookup



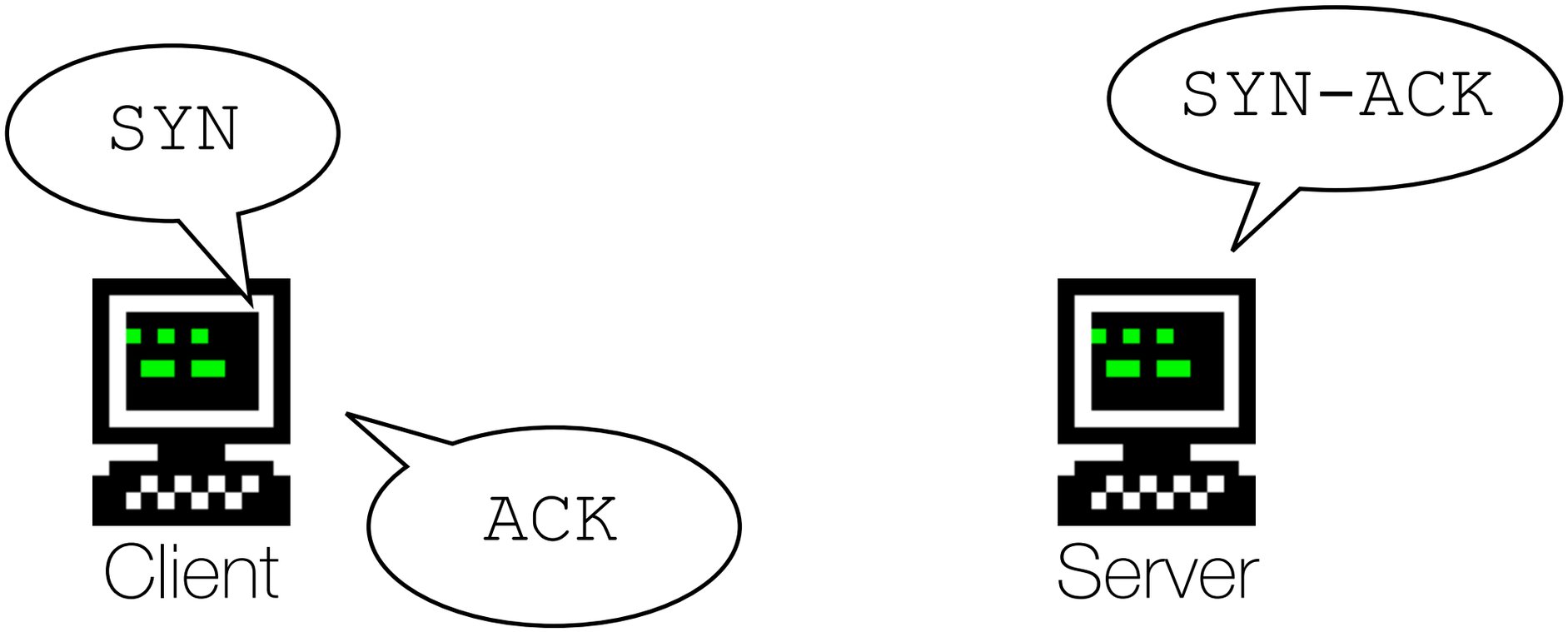
Browser hits DNS server which returns answer
(…after it asks a bunch more servers)

2. TCP Handshake



Client establishes TCP connection(s) with the server

2. TCP Handshake

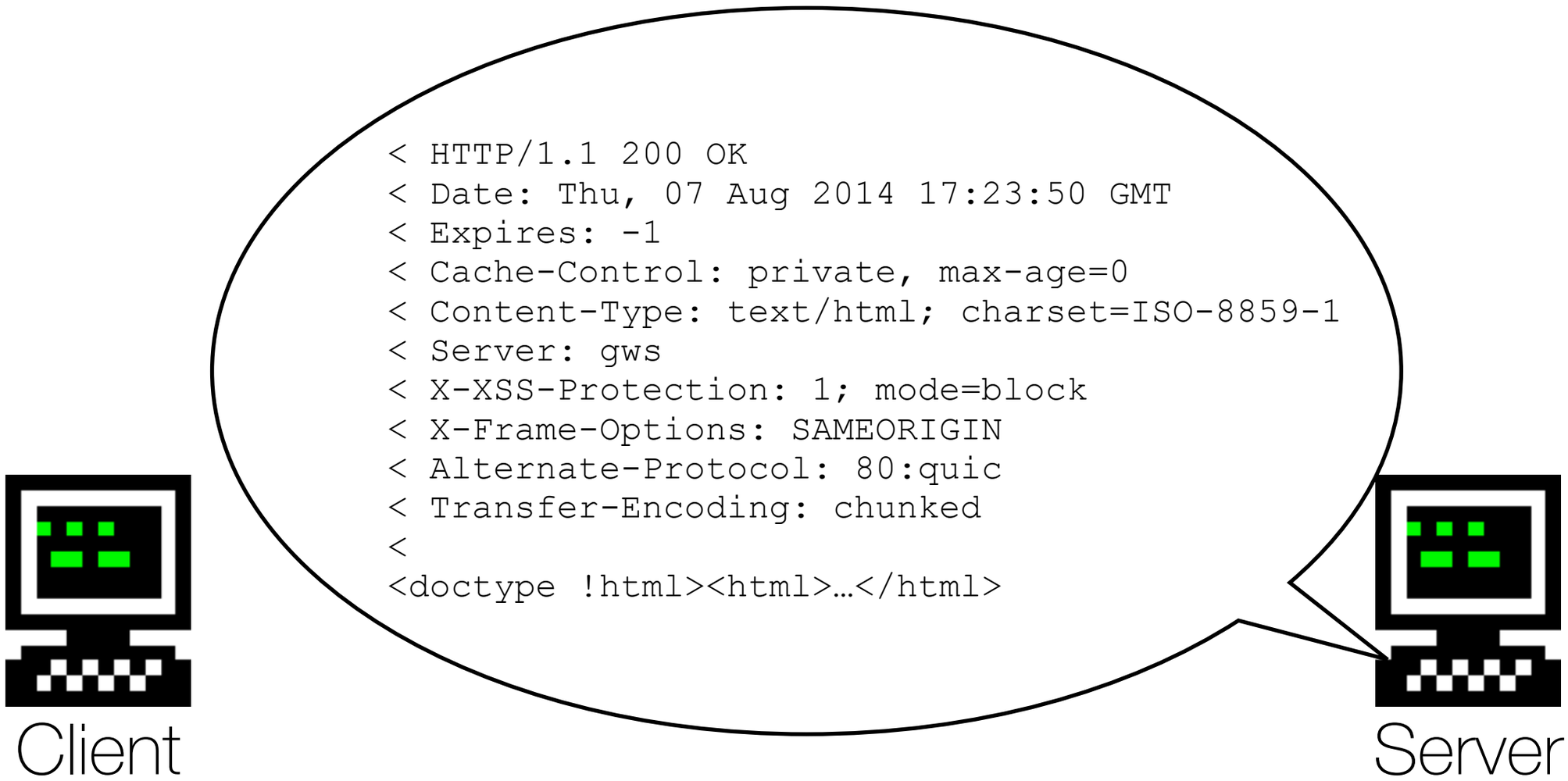


Client establishes TCP connection(s) with the server

3. HTTP request

Client sends HTTP request to server

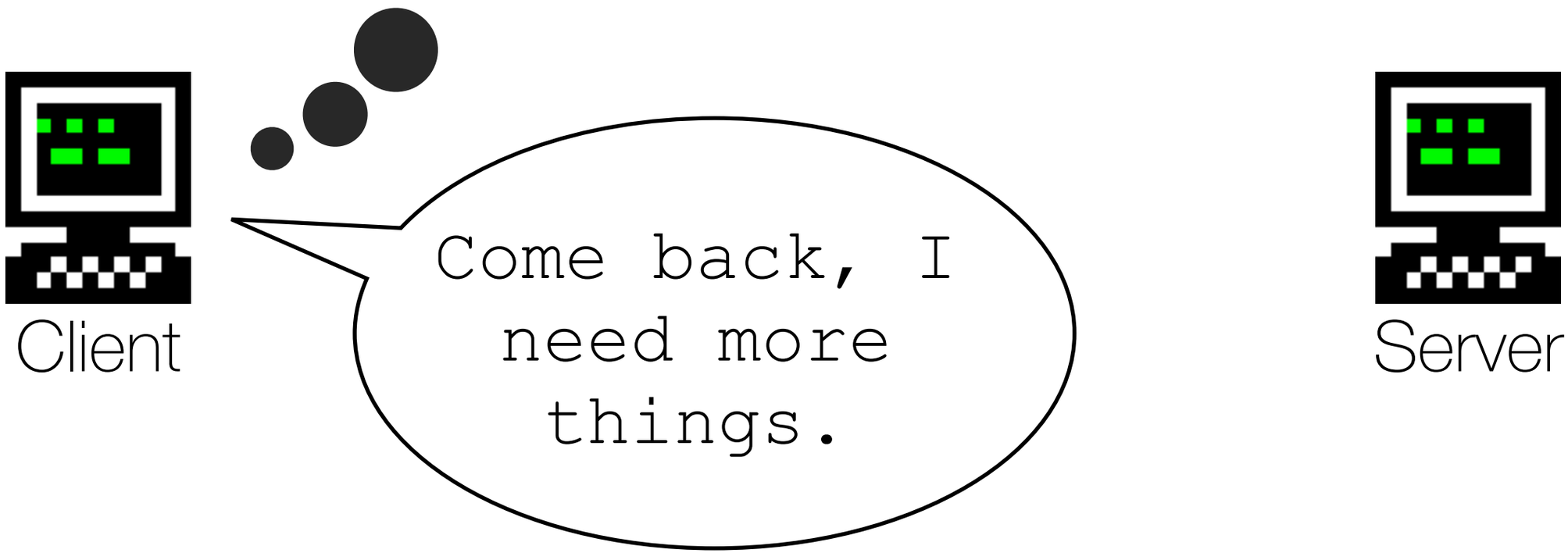
4. HTTP response



Server responds to client with HTTP response containing header followed by HTML document

5. HTML Parsing

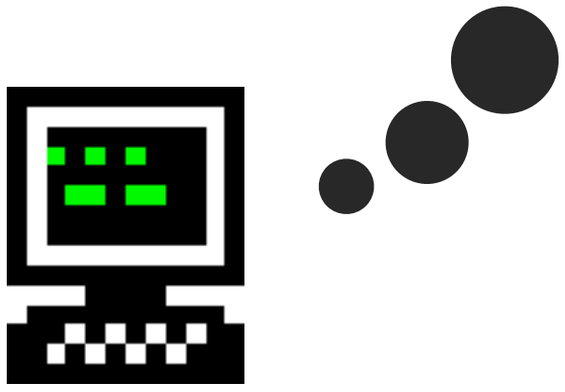
```
<!doctype html>
<html>
  <head>
    <script type="text/javascript" src="library.js"></script>
    <link rel="stylesheet" type="text/css" href="styles.css">
    <script type="text/javascript" src="website.js"></script>
  </head>
  <body>
    <div>Let's pretend this is all of google</div>
  </body>
</html>
```

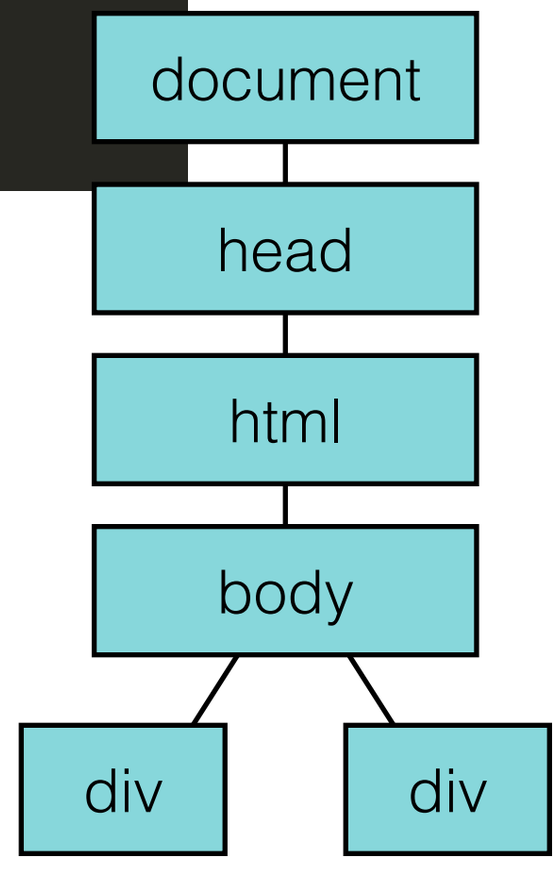


1. Speculative parser looks ahead for assets it can fetch

5. HTML Parsing

```
<!doctype html>
<html>
  <head>
    <script type="text/javascript" src="library.js"></script>
    <link rel="stylesheet" type="text/css" href="styles.css">
    <script type="text/javascript" src="website.js"></script>
  </head>
  <body>
    <div>Let's pretend this is all of google</div>
  </body>
</html>
```

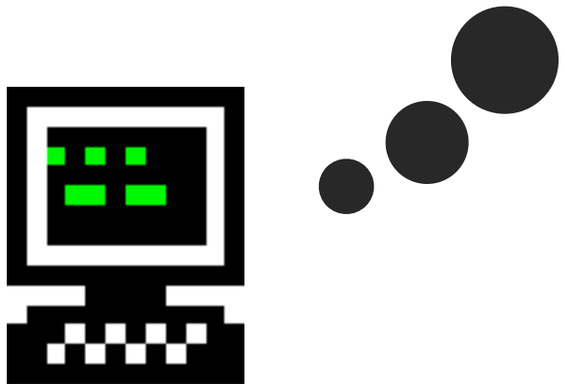




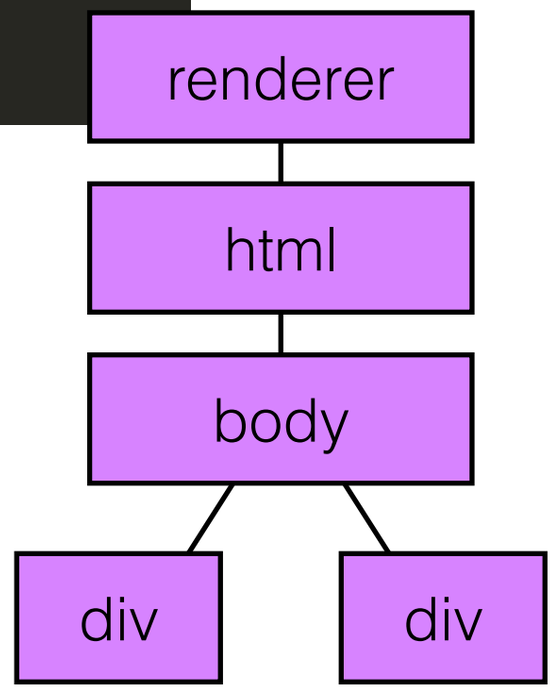
2. Main parser parses the HTML document, building a DOM tree out of the (likely broken…) HTML

5. HTML Parsing

```
<!doctype html>
<html>
  <head>
    <script type="text/javascript" src="library.js"></script>
    <link rel="stylesheet" type="text/css" href="styles.css">
    <script type="text/javascript" src="website.js"></script>
  </head>
  <body>
    <div>Let's pretend this is all of google</div>
  </body>
</html>
```

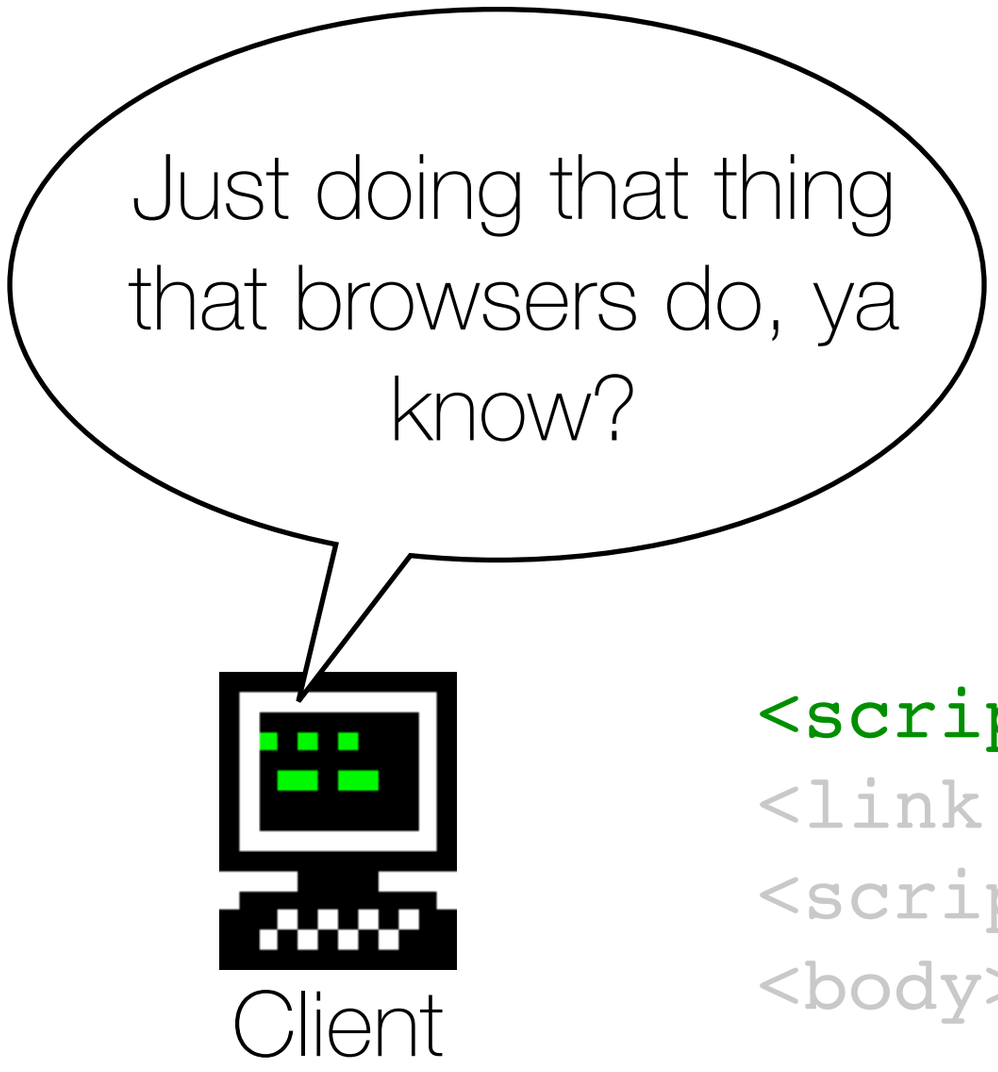


Client



3. Render tree is also made based on the DOM, containing only things that get rendered (so, not `<head>` or `display: none`)

5. HTML Parsing

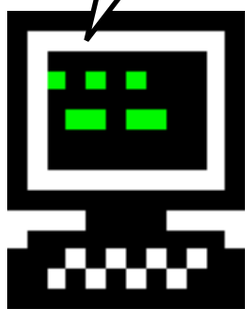



```
<script src="library.js"></script>
<link href="styles.css">
<script src="website.js"></script>
<body></body>
```

Client requests, parses, and executes all inlined assets (images, scripts, stylesheets).

5. HTML Parsing

Just doing that thing that browsers do, ya know?



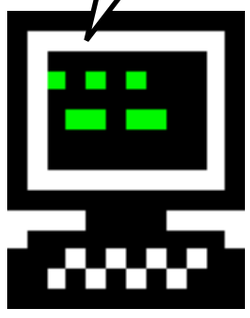
Client

```
<script async src="library.js"></script>
<link href="styles.css">
<script src="website.js"></script>
<body></body>
```

Client requests, parses, and executes all inlined assets (images, scripts, stylesheets).

5. HTML Parsing

Just doing that thing that browsers do, ya know?



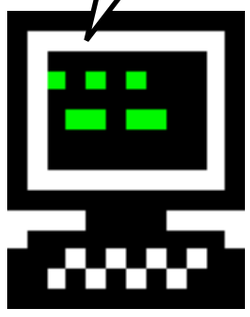
Client

```
<script defer src="library.js"></script>
<link href="styles.css">
<script src="website.js"></script>
<body></body>
```

Client requests, parses, and executes all inlined assets (images, scripts, stylesheets).

5. HTML Parsing

Just doing that thing that browsers do, ya know?



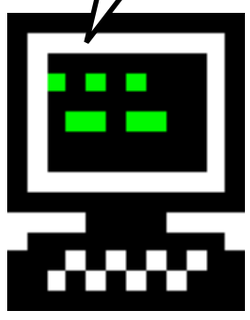
Client

```
<script src="library.js"></script>
<link href="styles.css">
<script src="website.js"></script>
<body></body>
```

Client requests, parses, and executes all inlined assets (images, scripts, stylesheets).

5. HTML Parsing

Just doing that thing that browsers do, ya know?



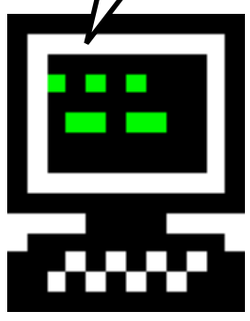
Client

```
<script src="library.js"></script>
<link href="styles.css">✓
<script src="website.js"></script>
<body></body>
```

Client requests, parses, and executes all inlined assets (images, scripts, stylesheets).

5. HTML Parsing

Just doing that thing that browsers do, ya know?



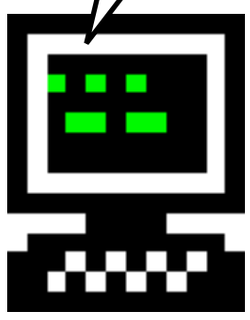
Client

```
<script src="library.js"></script> ✓
<link href="styles.css"> ✓
<script src="website.js"></script>
<body></body>
```

Client requests, parses, and executes all inlined assets (images, scripts, stylesheets).

5. HTML Parsing

Just doing that thing that browsers do, ya know?



Client

```
<script src="library.js"></script> ✓
<link href="styles.css"> ✓
<script src="website.js"></script> ✓
<body></body>
```

Client requests, parses, and executes all inlined assets (images, scripts, stylesheets).

5. HTML Parsing

```
1  <!doctype html>
2  <html>
3    <head>
4      <script type="text/javascript" src="library.js"></script>
5      <link rel="stylesheet" type="text/css" href="styles.css">
6      <script type="text/javascript" src="website.js"></script>
7    </head>
8    <body>
9      <div>Let's pretend this is all of google</div>
10   </body>
11 </html>
```

When client finishes parsing, DOMInteractive is fired and page is “ready”, and deferred scripts are downloaded, after which the load event is fired


THE INTERNET!!!!!