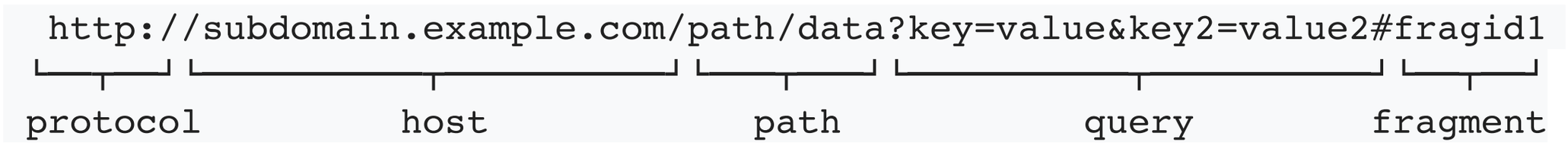
# Anatomy of a Page Request

Overview of the Drupal 8 Web Technology Stack
From browser to server and back

# Click a link, or type a URL in a browser and hit enter

http://govcontxewfcmvbo.devcloud.acquia-sites.com/about



# What happens next?

The URL is passed to your ISP
(possibly Comcast or Verizon) via your router

https://stackoverflow.com/questions/11887334/understanding-the-dns-lookup-mechnism

# Where is the domain hosted?

DNS lookup first runs through caches:

Browser
OS
ISP DNS Cache

# Server may send a redirect (301)

Which your browser will follow

https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

# Finally, an IP address is determined

64.194.136.5

# Drupal 8 server requirements

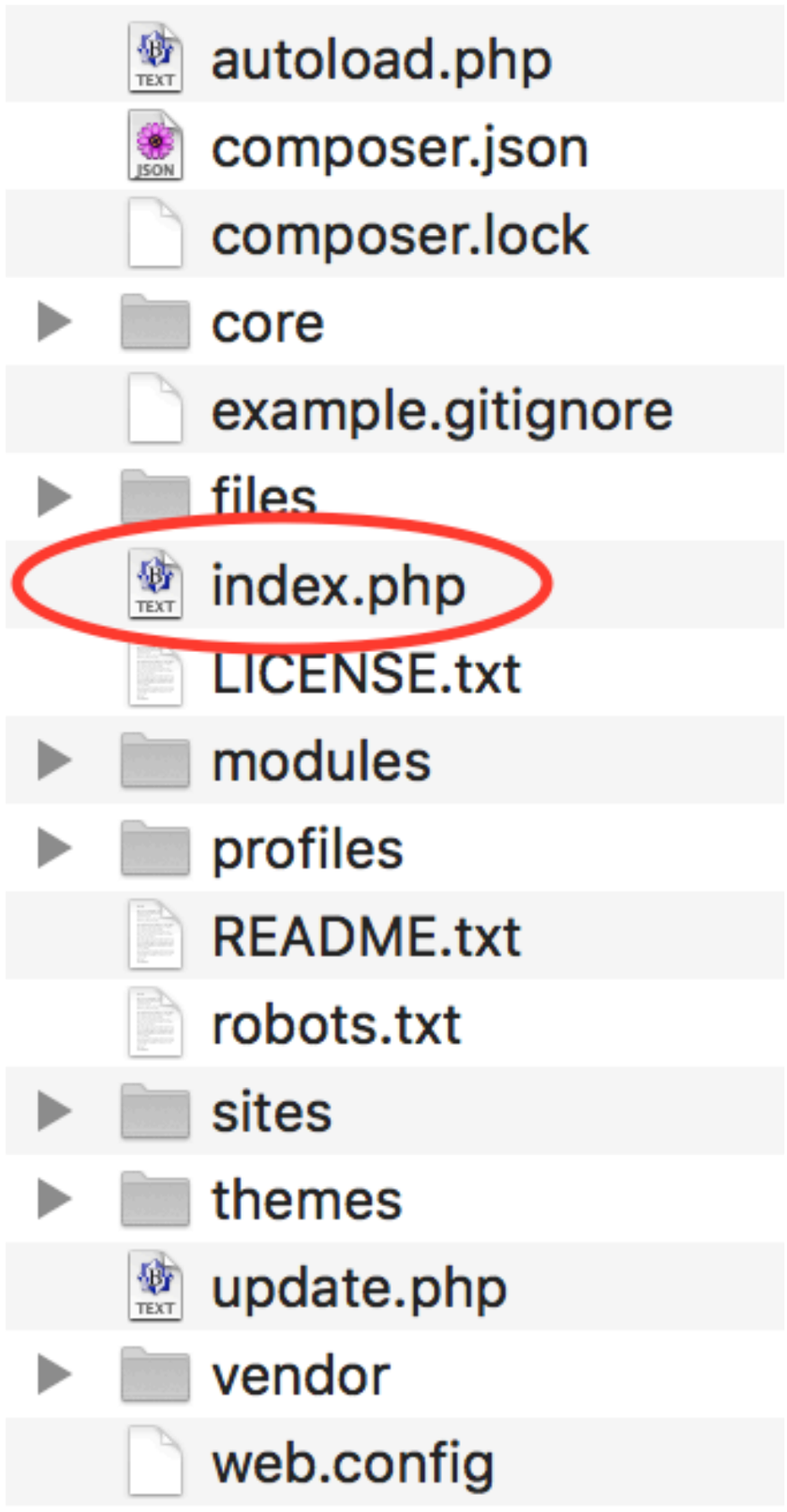
Operating system: Predominantly Linux
Apache HTTP server (or NGINX)
PHP (versions 5.6 or 7)
Database (MySQL/MariaDB/Percona)

# Web server software (Apache) receives the request and passes it to index.php

When configuring the web server, the document root directory is defined. DirectoryIndex sets the file that Apache will serve if that directory is requested

# index.php is found in the Drupal root directory

/var/www/html and /var/www/docroot are common directory structures in Linux


autoload.php
composer.json
composer.lock
core
example.gitignore
files
index.php
LICENSE.txt
modules
profiles
README.txt
robots.txt
sites
themes
update.php
vendor
web.config

```
<?php

/**
 * @file
 * The PHP page that serves all page requests on a Drupal
installation.
 *
 * All Drupal code is released under the GNU General Public
License.
 * See COPYRIGHT.txt and LICENSE.txt files in the "core" directory.
 */

use Drupal\Core\DrupalKernel;
use Symfony\Component\HttpFoundation\Request;

$autoloader = require_once 'autoload.php';

$kernel = new DrupalKernel('prod', $autoloader);

$request = Request::createFromGlobals();
$response = $kernel->handle($request);
$response->send();

$kernel->terminate($request, $response);
```

# Symfony creates the request

Provides the relevant HTTP functionality Drupal will need to deliver a response.

https://www.sitepoint.com/symfony-drupal-8/

# Autoload runs first

Autoload directs composer
to load PHP packages.

# DrupalKernel is created

Returns the appropriate site directory for a request and accesses the database specific to the site.

Drupal collects all the code it needs to create the page and loads it into server RAM.

# Drupal loads configuration from the database

From the database, Drupal finds the paths to the modules and themes it requires.

# The URL is compared for Aliases

The Path module allows you to specify a custom URL for any existing internal system path.

Relative URL: /admin/help/path

# Drupal parses the internal system path

Using the path provided in the $request, Drupal uses routing to match path arguments to functionality.

Routing determines which controllers defined in the DrupalKernel act on the response.

# Drupal loads content from the SQL database

Based on the content type, Drupal determines which fields to populate.

Based on the configuration, Drupal determines which other page elements are loaded

# Drupal kernel builds the Render(able) Array

Page content is assembled as an Associative array and passed to the theme.

An Associative array is a data structure that organizes elements into named key/value pairs.

https://www.drupal.org/node/2460415

# The Render(able) Array is converted to HTML

Using components of the Twig engine,
the Render(able) Array data is passed to templates
and converted to an HTML string.

The resulting HTML is the $response.

https://www.drupal.org/node/2460415

# The $response is returned.

Drupal returns the $response to Symfony and terminates the request. Symfony passes the response to Apache.

The $response is passed to your ISP, over the internet and back to your browser.

# It doesn’t end here

Next your browser to parses the HTML data and pulls in the CSS and JS files that are referenced.

# Convert the HTML string in the browser

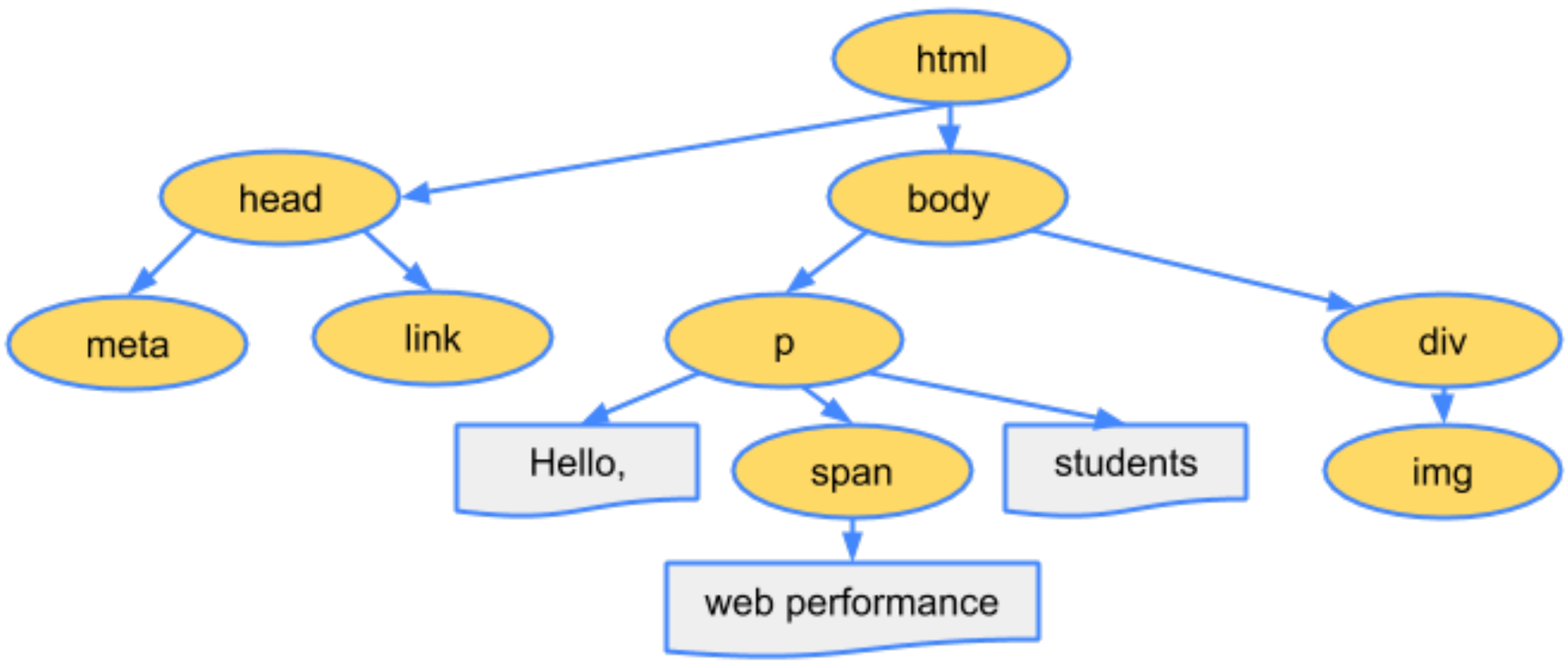
HTML markup: Document Object Model (DOM)

CSS markup: Cascading Style Sheet Object Model (CSSOM)

# Define nodes in the DOM

```
<html>
  <head>
    <meta name="viewport" content="width=device-width,initial-scale=1">
    <link href="style.css" rel="stylesheet">
    <title>Critical Path</title>
  </head>
  <body>
    <p>Hello <span>web performance</span> students!</p>
    <div><img src="awesome-photo.jpg"></div>
  </body>
</html>
```

# Define DOM tree structure




https://developers.google.com/web/fundamentals/performance/critical-rendering-path/constructing-the-object-model

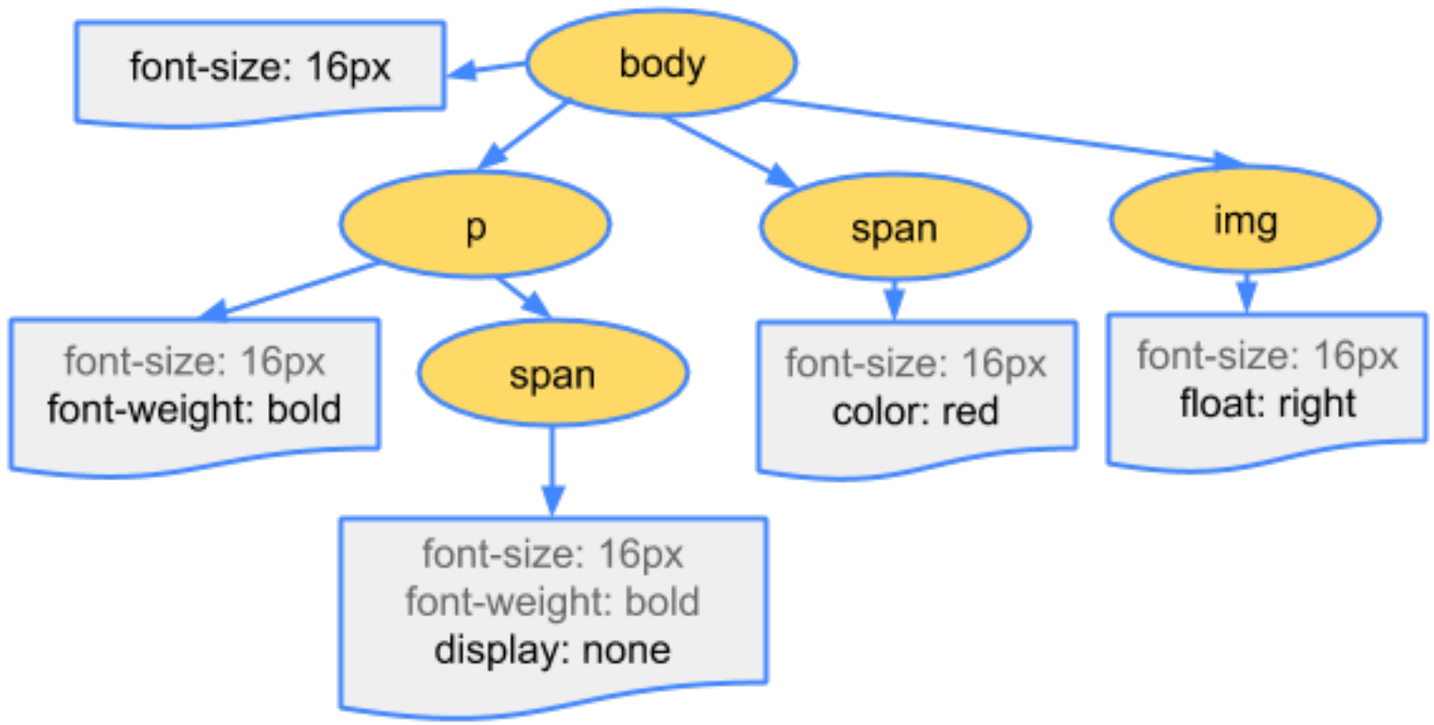
# Download CSS files

```
@import url("/core/assets/vendor/normalize-css/normalize.css?0");
@import url("/core/themes/stable/css/system/components/ajax-progress.module.css?0");
@import url("/core/themes/stable/css/system/components/align.module.css?0");
@import url("/core/themes/stable/css/system/components/autocomplete-loading.module.css?0");
@import url("/core/themes/stable/css/system/components/fieldgroup.module.css?0");
@import url("/core/themes/stable/css/system/components/container-inline.module.css?0");
@import url("/core/themes/stable/css/system/components/clearfix.module.css?0");
```

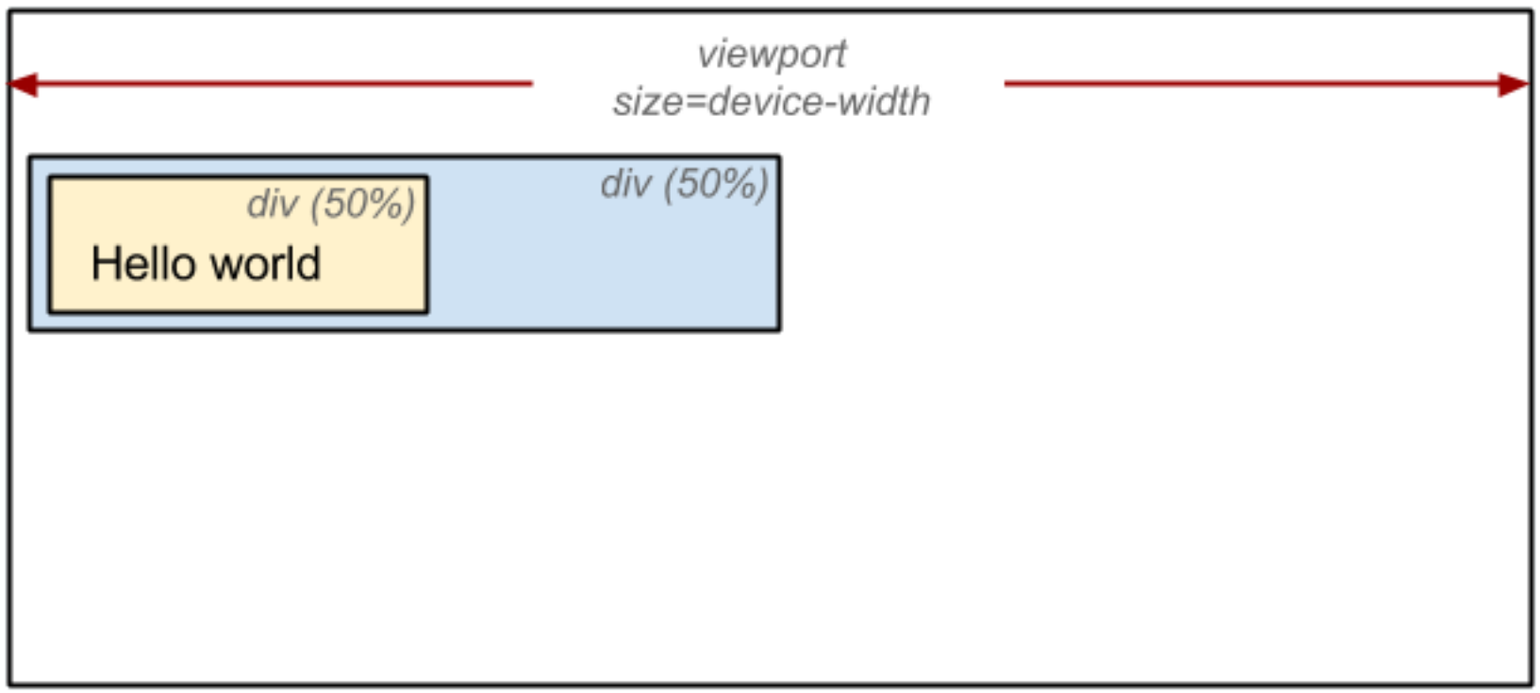
# A Cascading Style Sheet

```
body { font-size:
16px }
p { font-weight:
bold }
span { color: red }
p span { display:
none }
img { float:
right }
```

# CSS applied to CSSOM



# Render, Layout and Paint



# Download JavaScript files

```
<script src="/core/assets/vendor/jquery/jquery.min.js?v=2.2.4"></script>
<script src="/core/assets/vendor/underscore/underscore-min.js?v=1.8.3"></script>
<script src="/core/assets/vendor/backbone/backbone-min.js?v=1.2.3"></script>
<script src="/core/assets/vendor/jquery-once/jquery.once.min.js?v=2.1.1"></script>
<script src="/core/misc/drupalSettingsLoader.js?v=8.3.5"></script>
<script src="/core/misc/drupal.js?v=8.3.5"></script>
<script src="/core/misc/drupal.init.js?v=8.3.5"></script>
<script src="/core/assets/vendor/jquery.ui/ui/core-min.js?v=1.11.4"></script>
```

# JavaScript Supports Interactivity

```
  <script>
      var span = document.getElementsByTagName('span')[0];
      span.textContent = 'interactive'; // change DOM text
content
      span.style.display = 'inline';  // change CSSOM property
      // create a new element, style it, and append it to the DOM
      var loadTime = document.createElement('div');
      loadTime.textContent = 'You loaded this page on: ' + new
Date();
      loadTime.style.color = 'blue';
      document.body.appendChild(loadTime);
  </script>
```

# Profit!

Anatomy of a Page Request

Charles Novick

Drupal GovCon 2017