



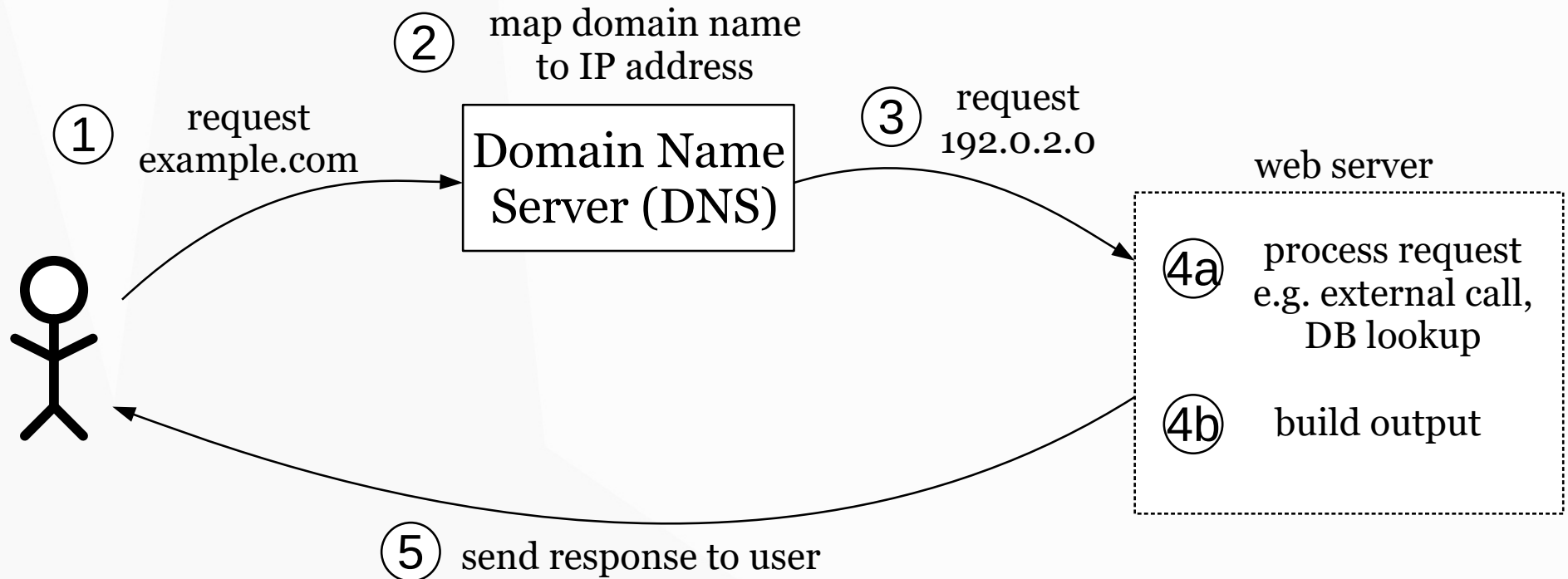
Building Static Sites with Hugo and GitHub

Charles Lee

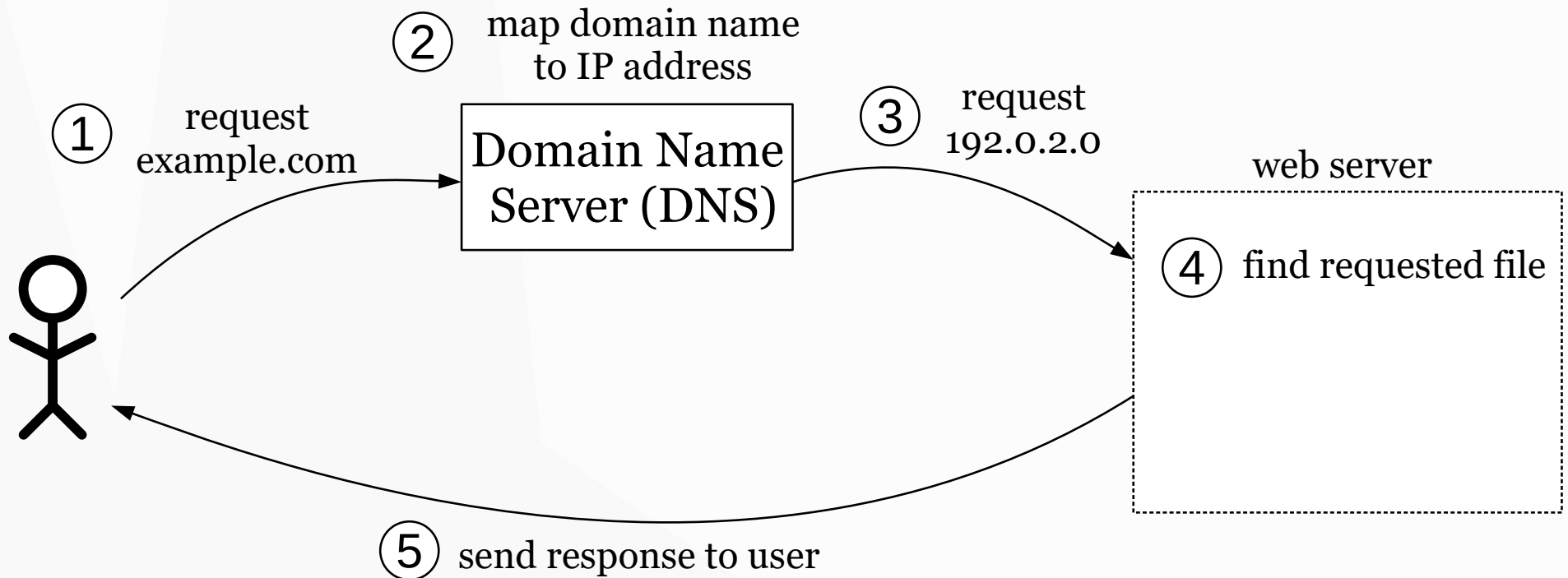
Agenda

- 1) What are static websites?
- 2) How to make static sites using Hugo?
- 3) How to host static sites using GitHub?

Typical (aka dynamic) site workflow



Static site workflow



Pros & Cons of Static Sites

	Dynamic	Static
Pros	Support arbitrary complexity Standard configuration	Simpler setup and configuration Faster response time Cheaper/free hosting
Cons	More complexity (failure modes) More expensive hosting	Can't serve dynamic content Immature tooling Requires technical expertise
Good use cases	Social interaction (e.g. forums, profiles, messaging) Client-server interaction (e.g. Google Maps)	Personal blog/portfolio Company/product website Documentation

Evolution towards Static Site Generators (SSG)

Maintain manually (naive)

- every file separate
- problem: changing common features

Maintain manually (modular)

- extract shared components
- problem: cross-referencing, content coupled with design

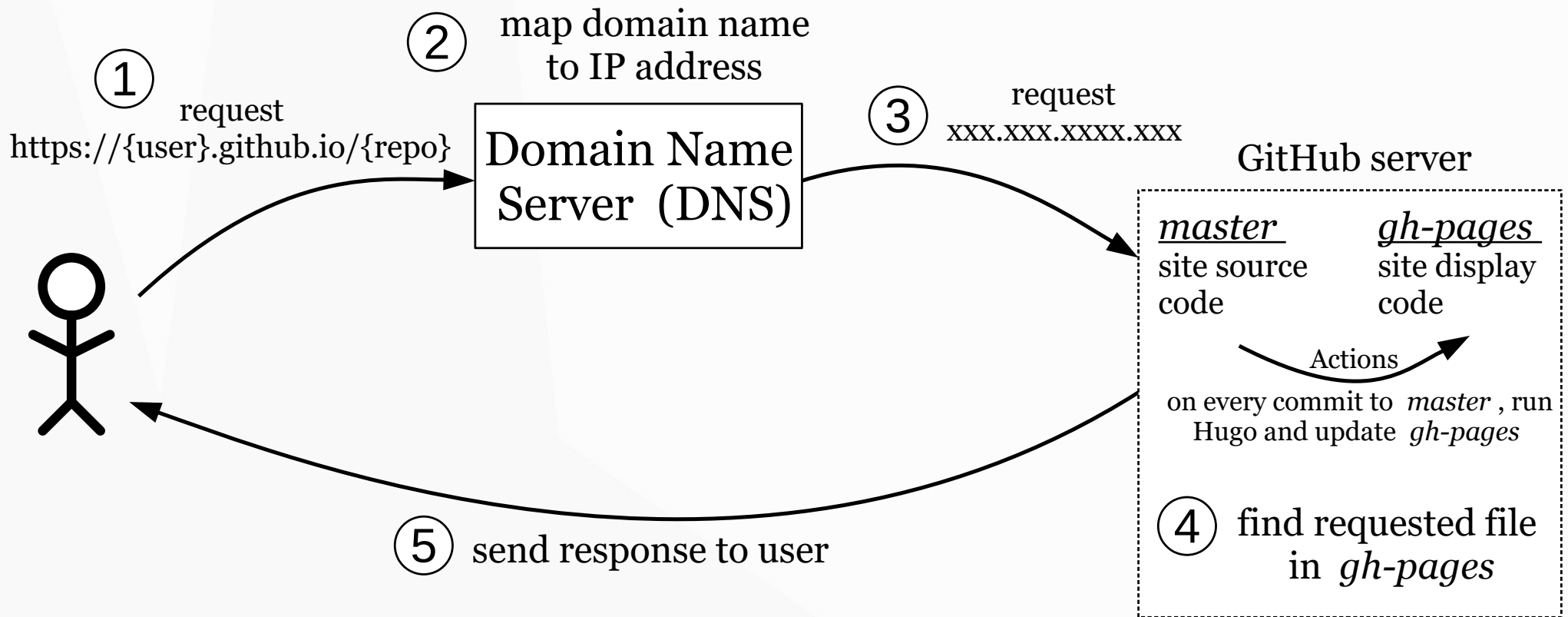
Use SSG

- Hugo (<1ms/page, “simple”)
- problem: learn new system

Write some code

- pre-process: source --> display
- placeholders for shared components
- templates for new objects
- problem: features, maintenance

Static Site workflow with GitHub



Demo time

- 1) Clone and configure starting repo
 - <https://github.com/charlesjlee/hugo-github-pages-actions-demo>
- 2) Create more posts
- 3) Switch themes
- 4) Override template

More Hugo features

- Dynamic-like content
 - Calls to 3rd party services, e.g. [ecommerce](#), [disqus](#)
 - Client-side Javascript, e.g. [search](#)
 - Data-driven content via [getJSON](#), [getCSV](#)

```
<ul>
  {{ $urlPre := "https://api.github.com" }}
  {{ $gistJ := getJSON $urlPre "/users/GITHUB_USERNAME/gists" }}
  {{ range first 5 $gistJ }}
    {{ if .public }}
      <li><a href="{{ .html_url }}" target="_blank">{{ .description }}</a></li>
    {{ end }}
  {{ end }}
</ul>
```

- Internationalization and localization (i18n) [support](#)
- Combine and layer themes via [components](#)