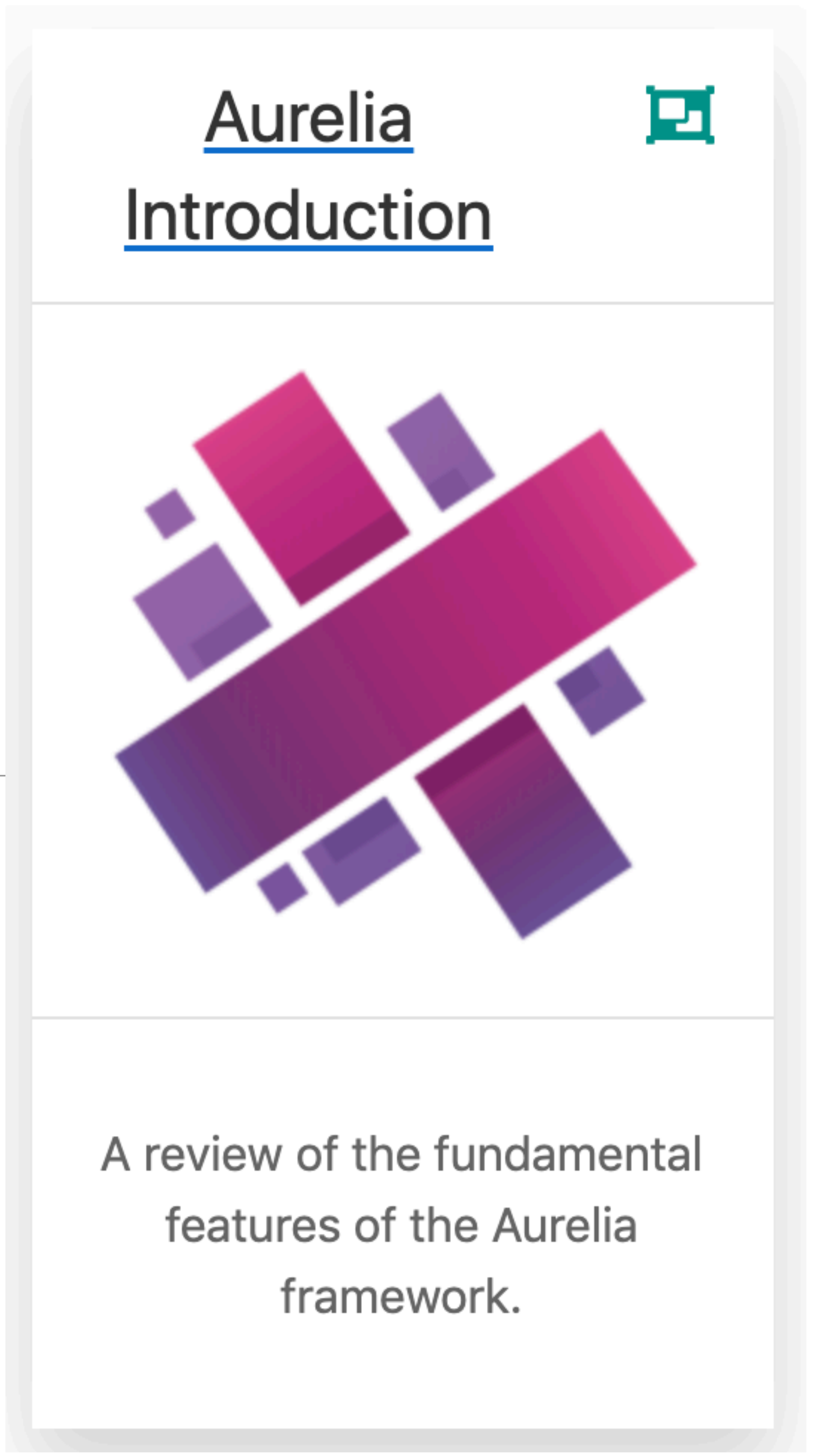
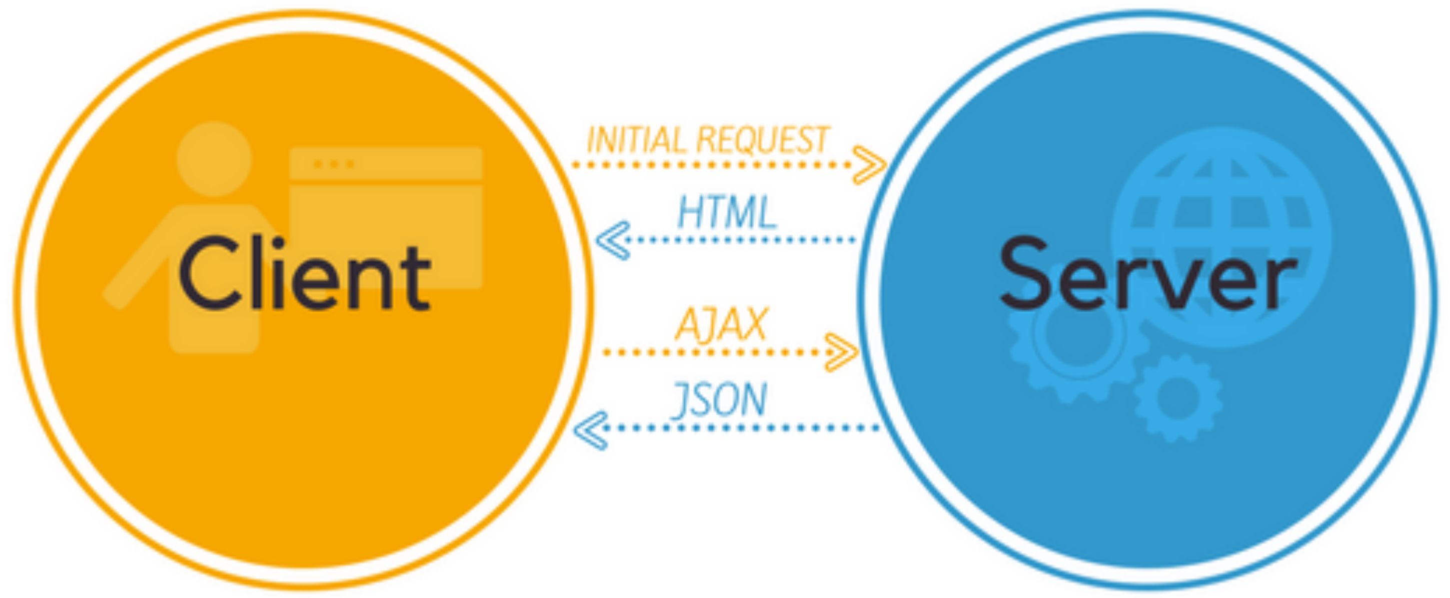
Introducing Aurelia



Aurelia Introduction

A review of the fundamental features of the Aurelia framework.

Single Page Applications

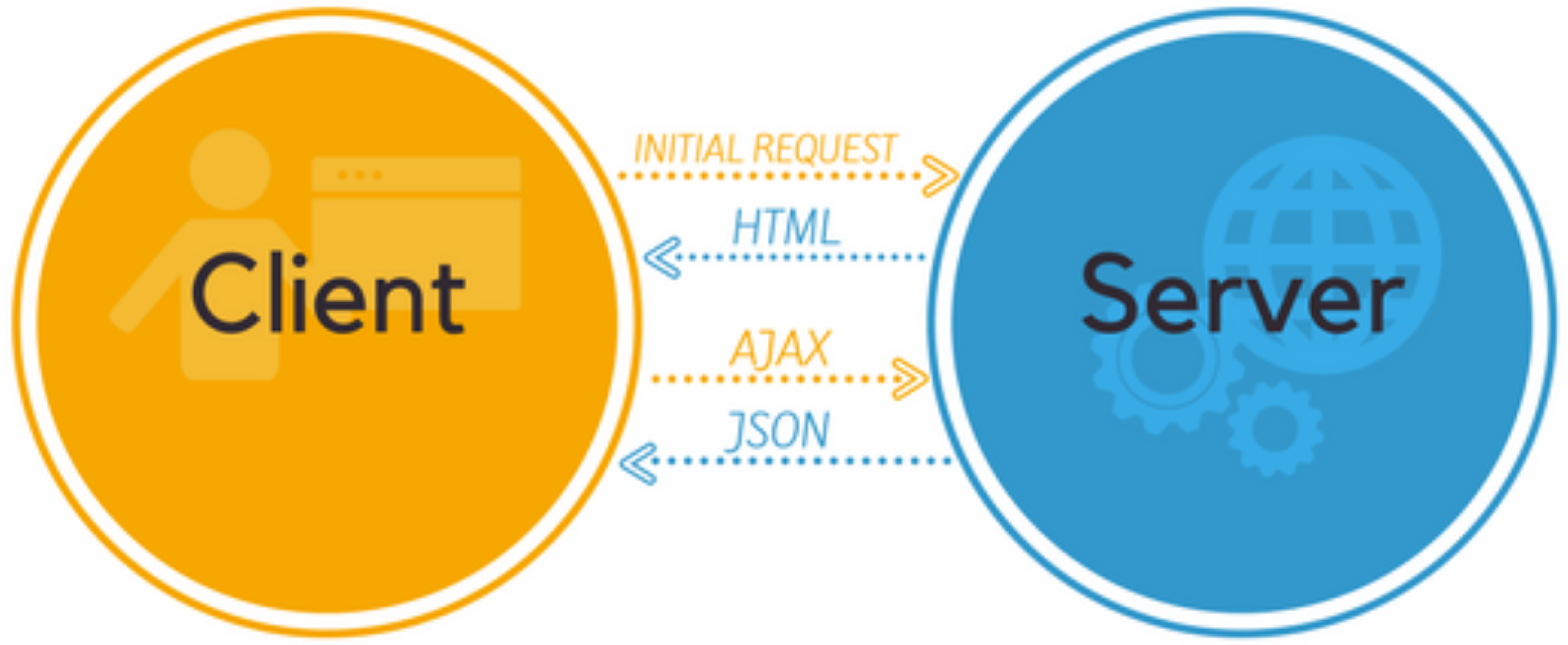


Web Components

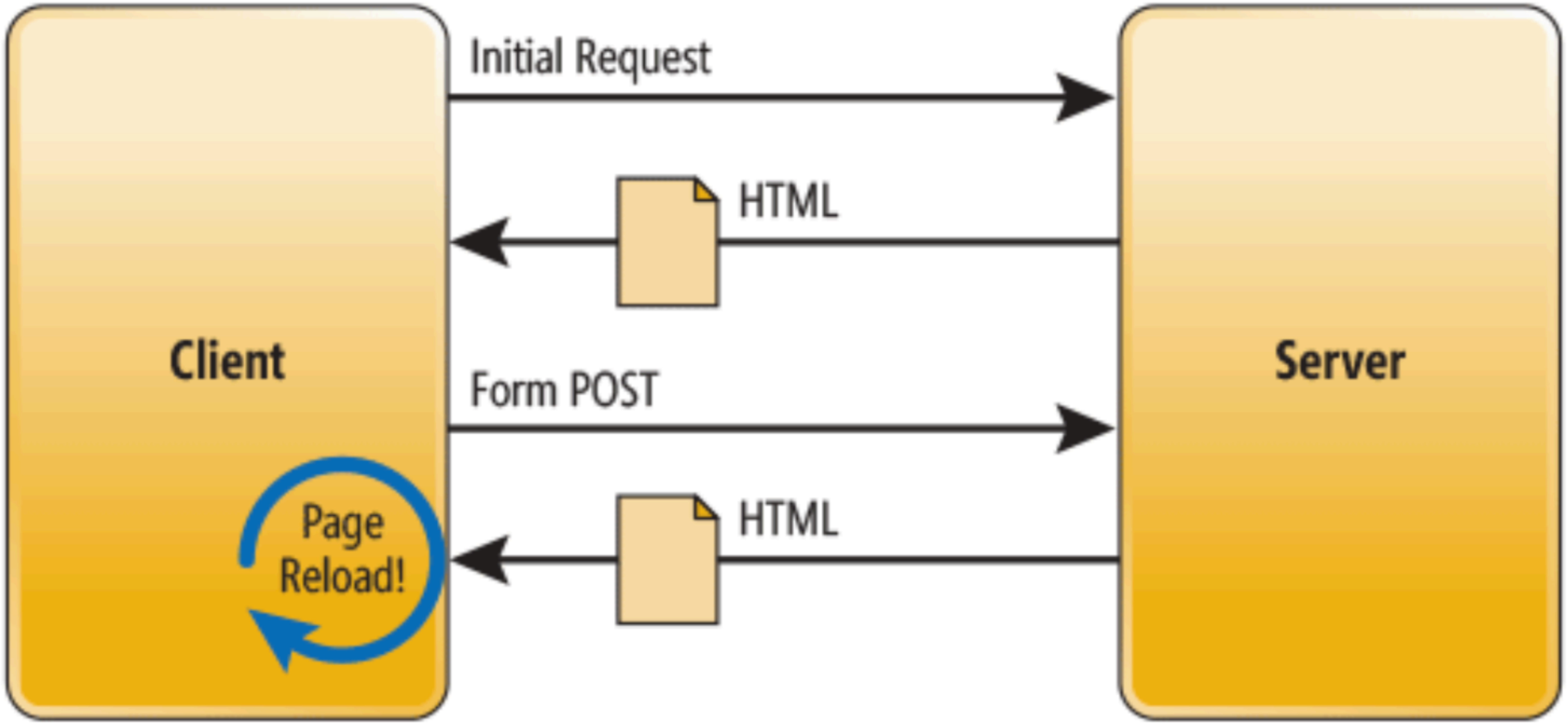
Aurelia

Agenda

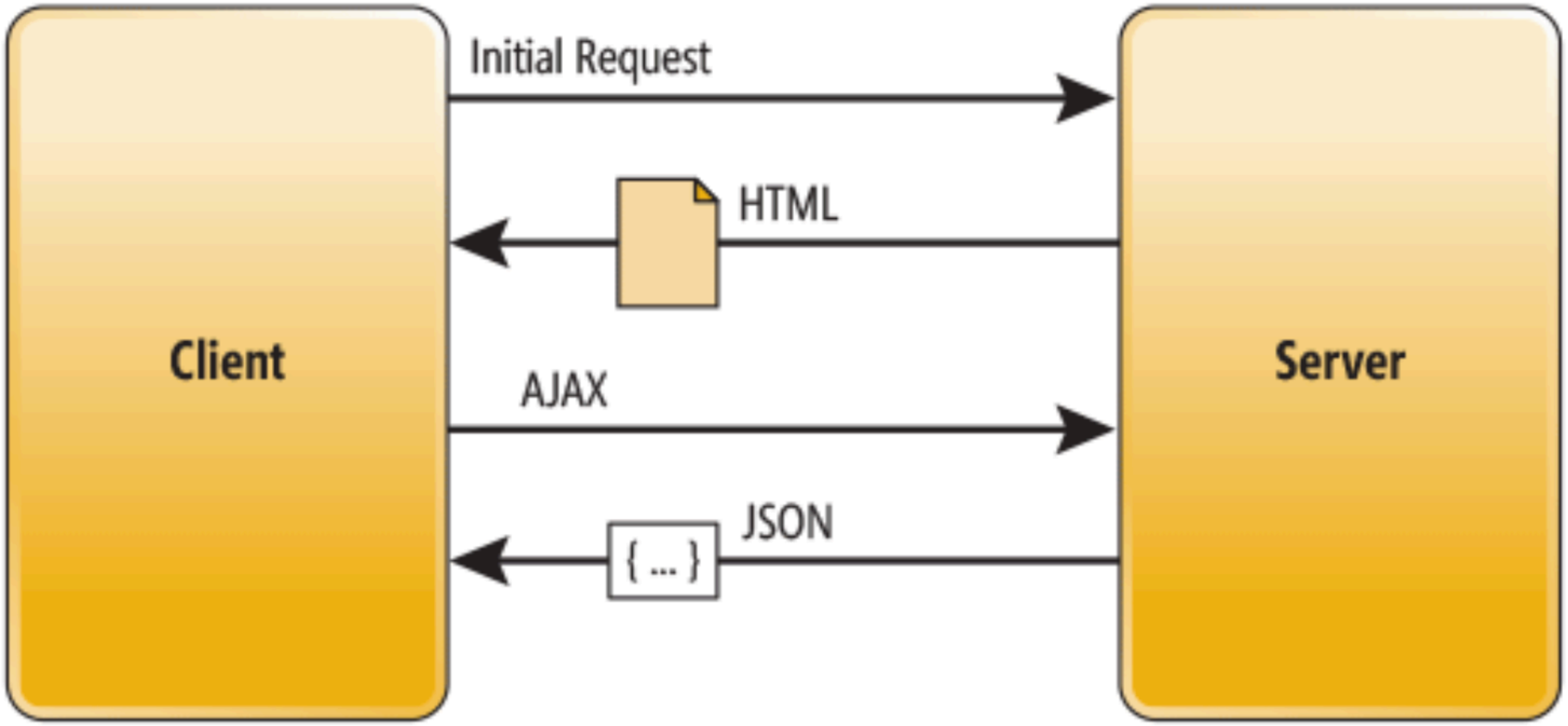
Single Page Applications



Traditional Page Lifecycle

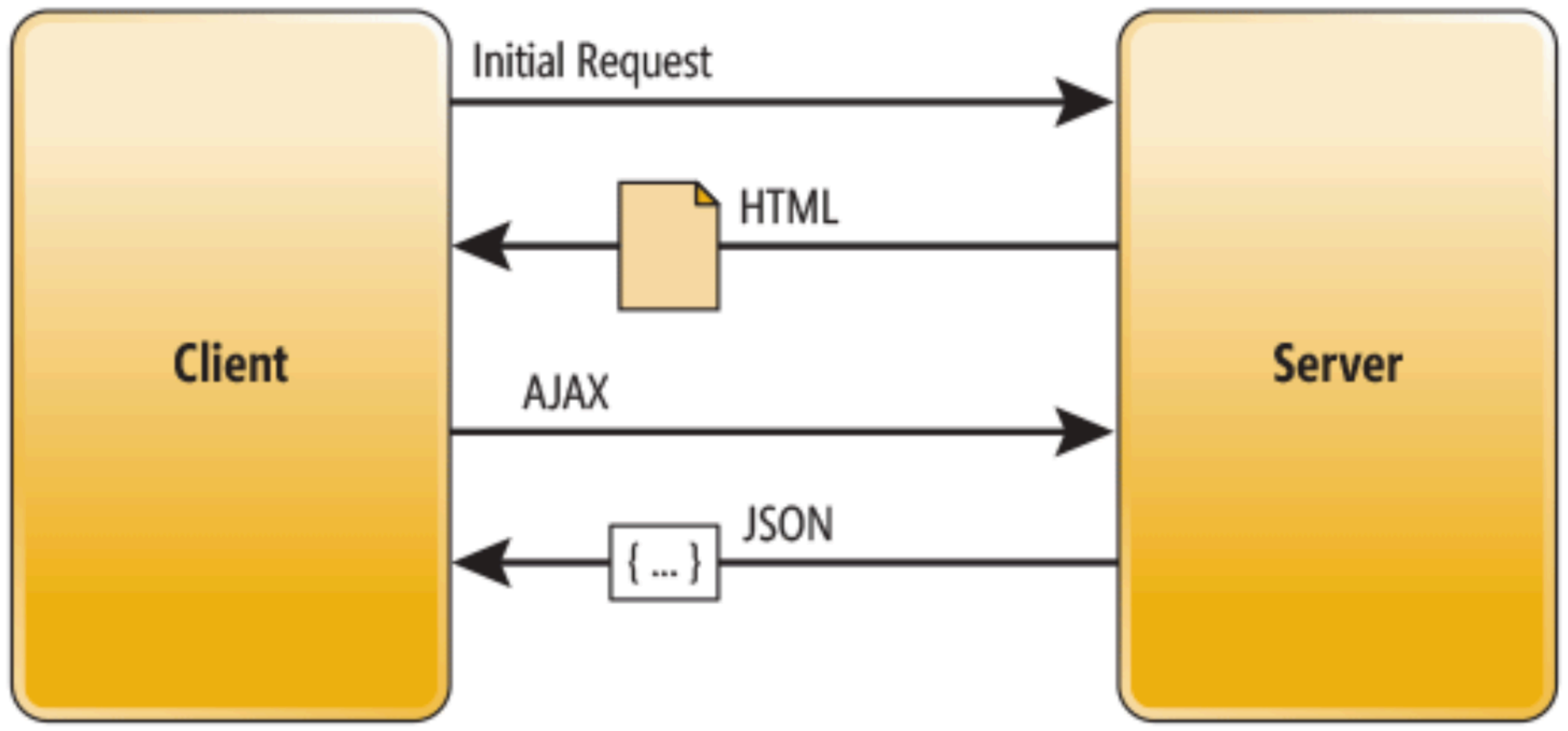


Single Page Application Lifecycle



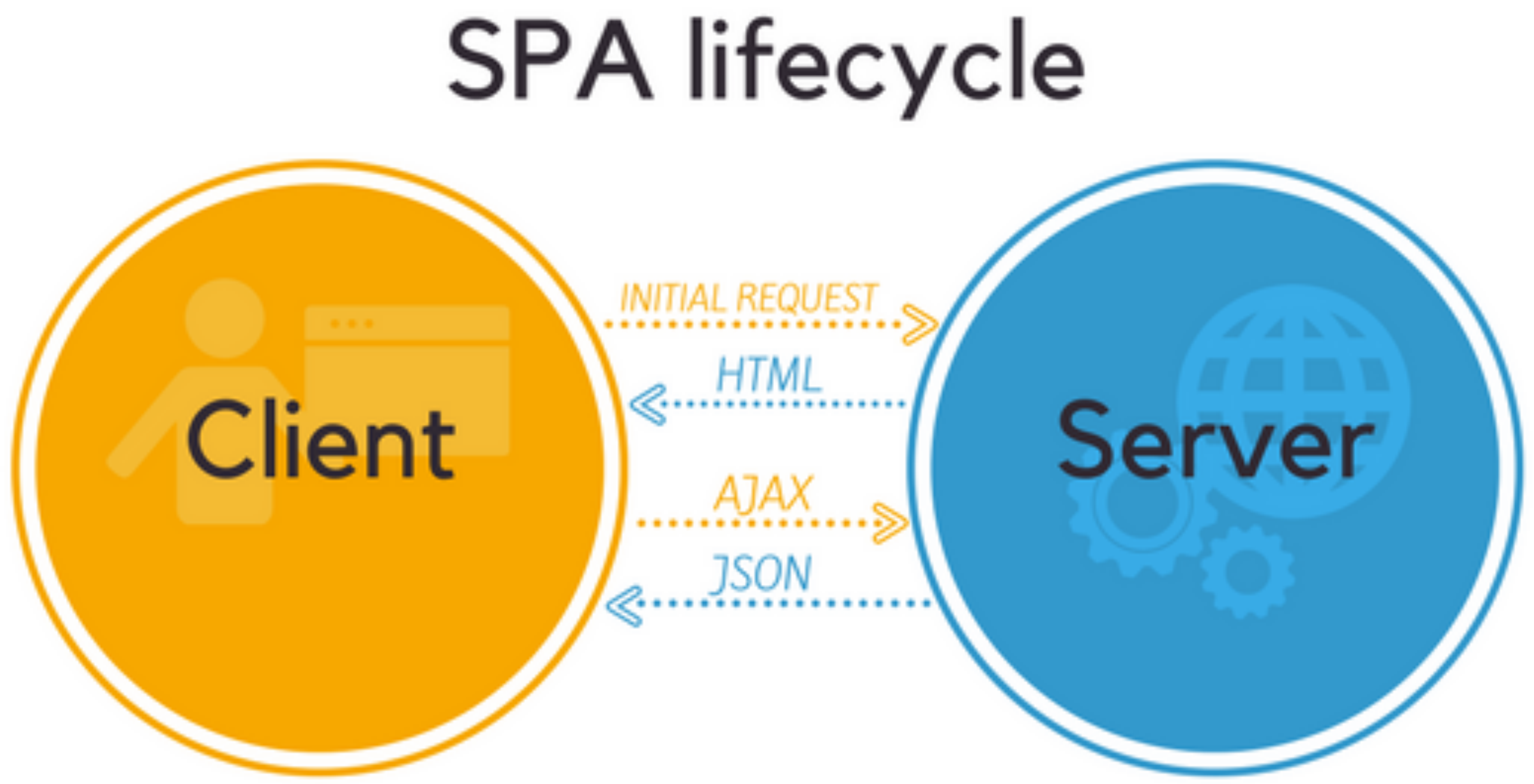
Single Pages Apps (SPAs)

- Single Page Applications (SPAs) are a web app served up as a single HTML request.
  - One initial page load of HTML
  - Dynamic features via sophisticated Javascript/ AJAX incorporated into the page
- Built with a client-side MVC (Model View Controller)/ MVVM (Model View View-Model) library or framework (Angular, Ember, Aurelia, React, Vue)
- Interact with Rest backends - using JSON default data format



Key Features of SPAs

- Back-end language agnostic
- Apps usually driven by data and events
- Enhanced Performance & User Experience
- Decoupling/testability
- Easier to build/maintain
- Heavy JS lifting on the client, lighter back-end
- Easier to provide offline operation



SPA - Performance

- Load time - One file each of HTML, CSS, JS, static files not dynamic
- Less data transfer: XHR calls only send raw data, not HTML markup
- Load distribution: dramatically less load on server, by distributing it to clients

SPA - UX

- AJAX and SPAs have raised the bar for user expectations
- Besides actually being faster, JS interactions make apps feel more responsive
- Immediate feedback on click
- Smaller data transfer means faster responses

Web Components

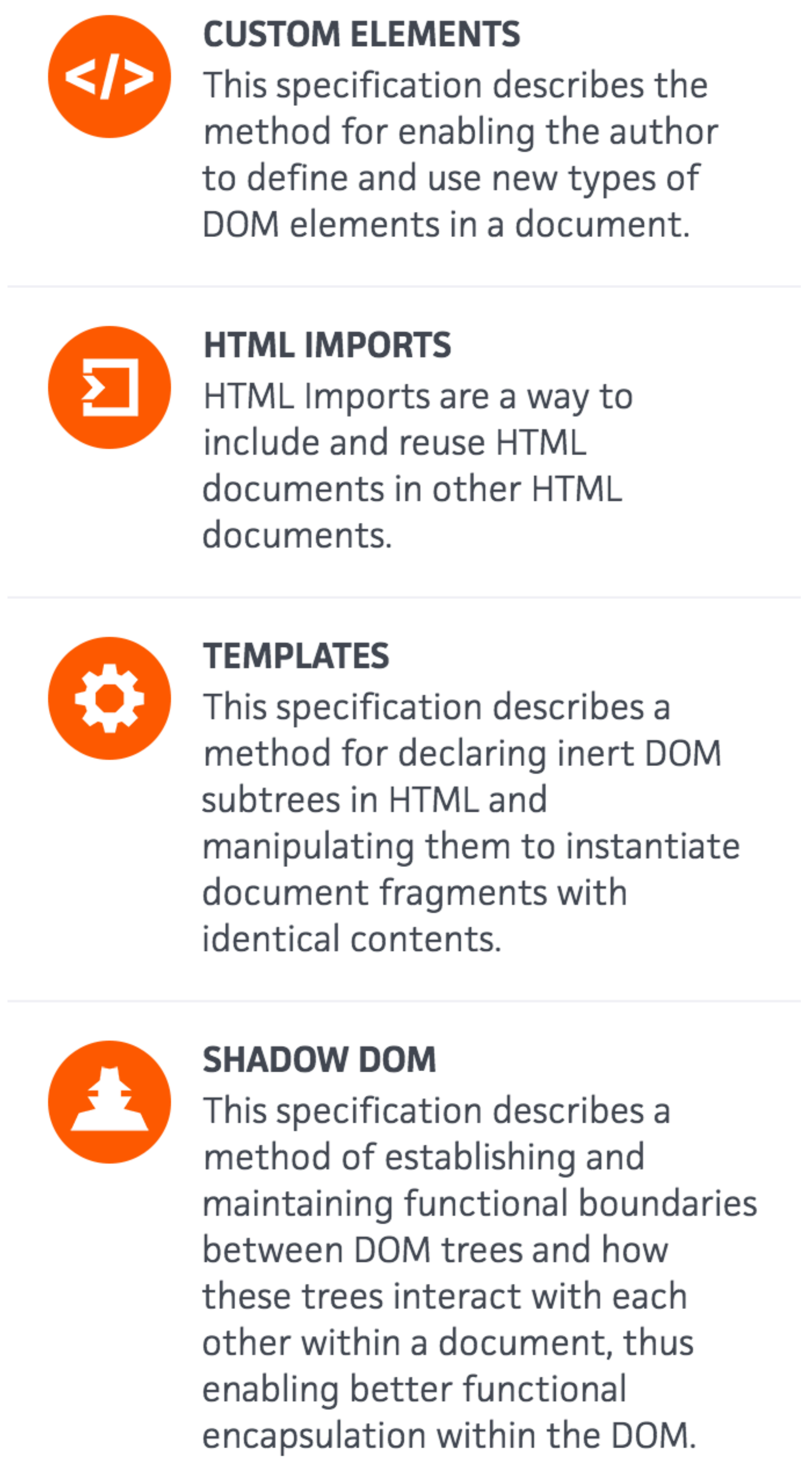
Web Components

- A set of features currently being added by the W3C to the HTML and DOM specifications that allow for the creation of reusable components in web applications.
- Bring component-based software engineering to the World Wide Web.
- The components model allows for encapsulation and interoperability of individual HTML elements.

Web Components

- An emerging standard for the web development

WebComponents.org



CUSTOM ELEMENTS
This specification describes the method for enabling the author to define and use new types of DOM elements in a document.

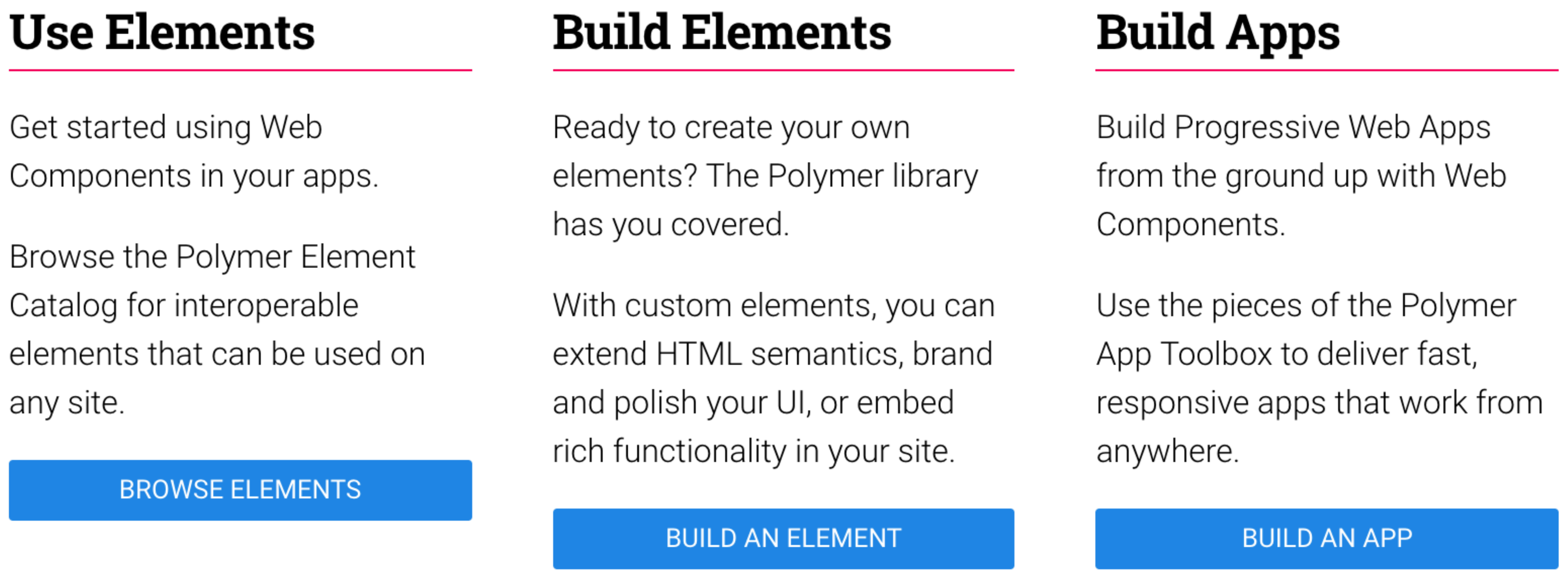
HTML IMPORTS
HTML Imports are a way to include and reuse HTML documents in other HTML documents.

TEMPLATES
This specification describes a method for declaring inert DOM subtrees in HTML and manipulating them to instantiate document fragments with identical contents.

SHADOW DOM
This specification describes a method of establishing and maintaining functional boundaries between DOM trees and how these trees interact with each other within a document, thus enabling better functional encapsulation within the DOM.

Polymer Web Components Library

- Google project to pioneer Web Component Libraries



Use Elements

Get started using Web Components in your apps.

Browse the Polymer Element Catalog for interoperable elements that can be used on any site.

Build Elements

Ready to create your own elements? The Polymer library has you covered.

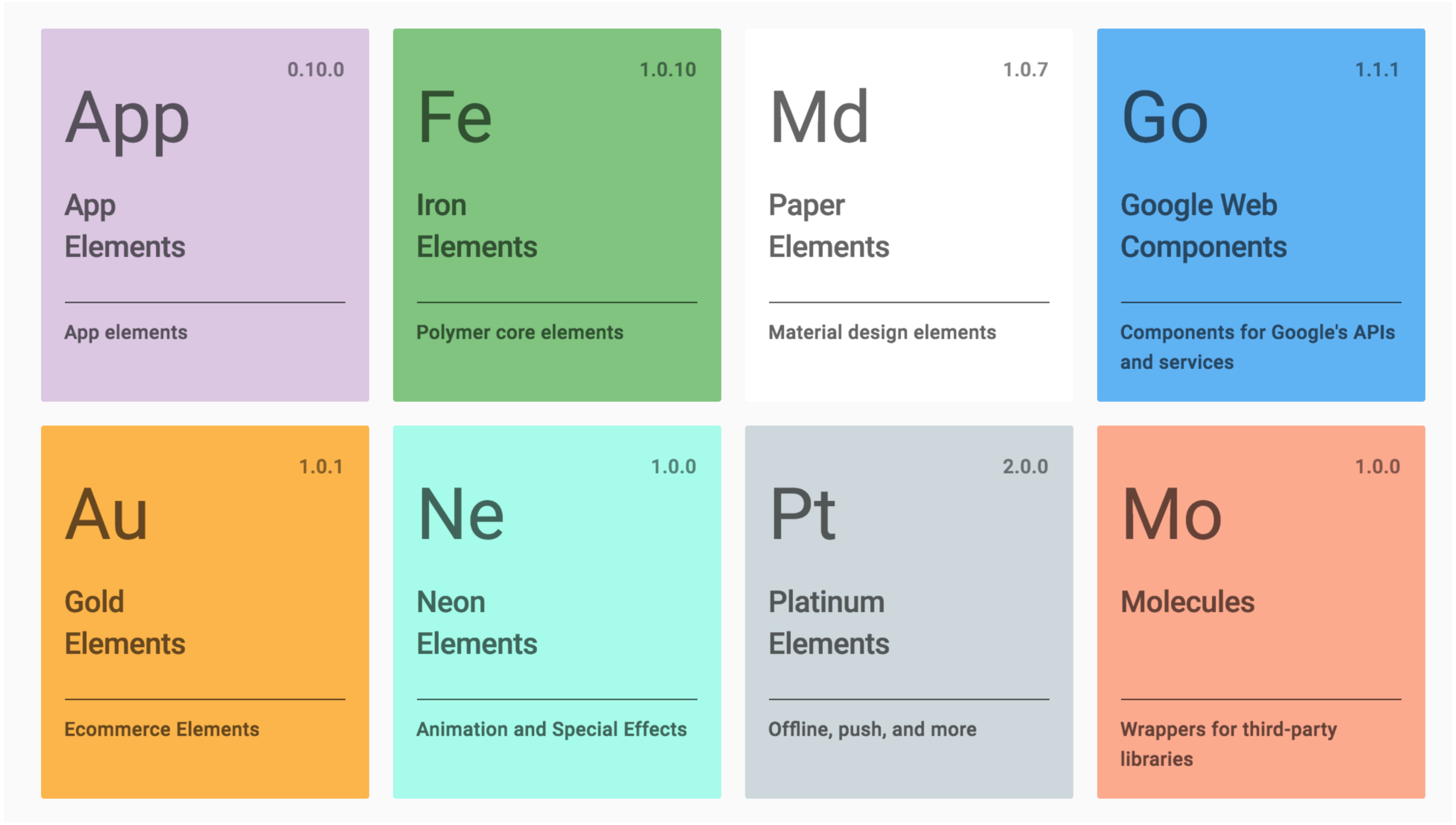
With custom elements, you can extend HTML semantics, brand and polish your UI, or embed rich functionality in your site.

Build Apps

Build Progressive Web Apps from the ground up with Web Components.

Use the pieces of the Polymer App Toolbox to deliver fast, responsive apps that work from anywhere.

Polymer Catalogue



0.10.0
App
App Elements
App elements

1.0.10
Fe
Iron Elements
Polymer core elements

1.0.7
Md
Paper Elements
Material design elements

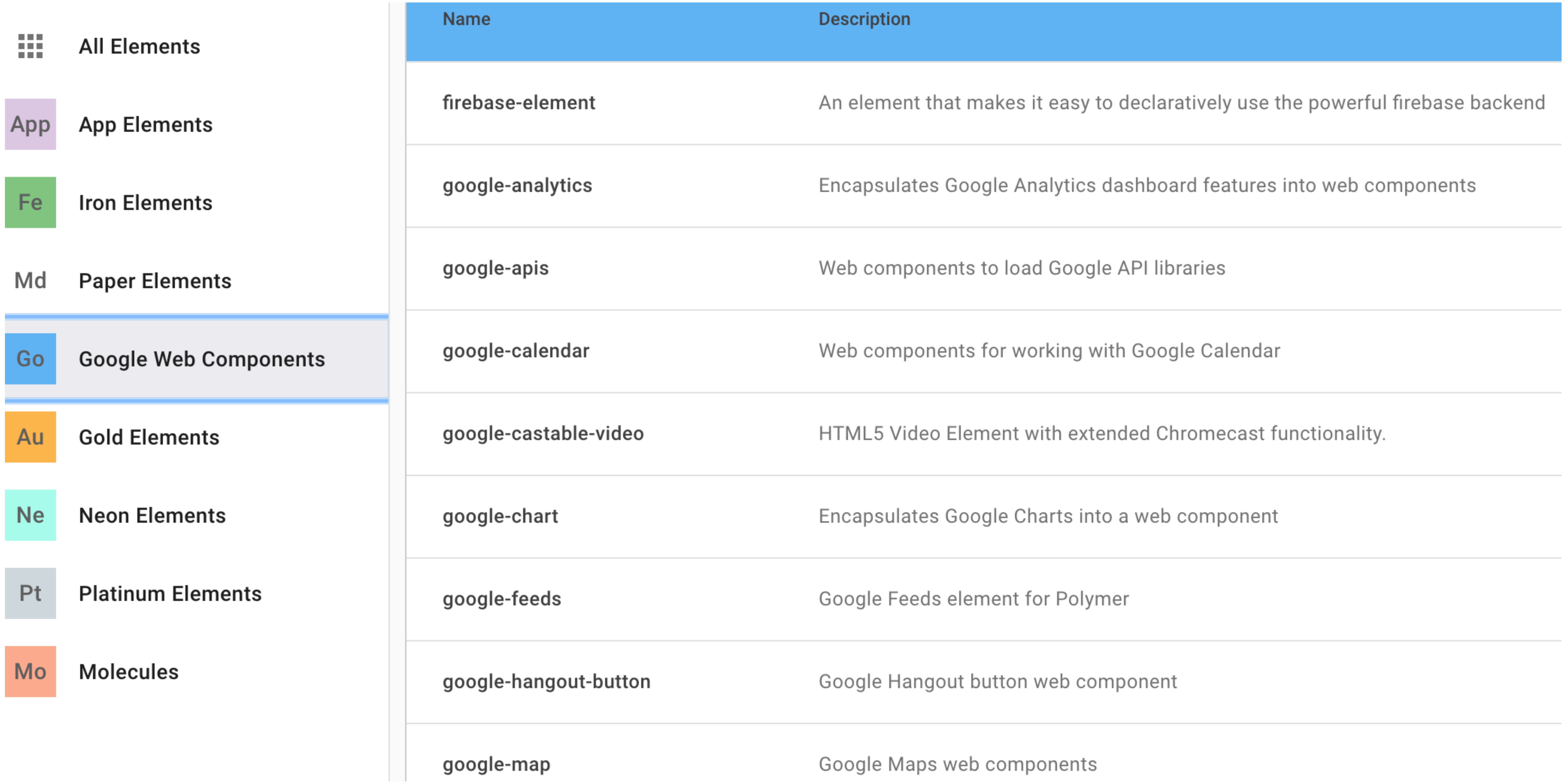
1.1.1
Go
Google Web Components
Components for Google's APIs and services

1.0.1
Au
Gold Elements
Ecommerce Elements

1.0.0
Ne
Neon Elements
Animation and Special Effects

2.0.0
Pt
Platinum Elements
Offline, push, and more

1.0.0
Mo
Molecules
Wrappers for third-party libraries



▦	All Elements
App	App Elements
Fe	Iron Elements
Md	Paper Elements
Go	Google Web Components
Au	Gold Elements
Ne	Neon Elements
Pt	Platinum Elements
Mo	Molecules

Name	Description
firebase-element	An element that makes it easy to declaratively use the powerful firebase backend
google-analytics	Encapsulates Google Analytics dashboard features into web components
google-apis	Web components to load Google API libraries
google-calendar	Web components for working with Google Calendar
google-castable-video	HTML5 Video Element with extended Chromecast functionality.
google-chart	Encapsulates Google Charts into a web component
google-feeds	Google Feeds element for Polymer
google-hangout-button	Google Hangout button web component
google-map	Google Maps web components

Aurelia

Aurelia

“Aurelia is a JavaScript client framework for mobile, desktop and web leveraging simple conventions and empowering creativity”

Technical Principles:

- Forward-thinking
- Two-Way Databinding
- Routing & UI Composition
- Broad Language Support
- Modern Architecture
- Extensible HTML
- MV* with Conventions
- Testable

Forward-thinking

- Written with next-generation Javascript OR Typescript.
- Integrates with Web Components.
- No external dependencies.
- Leverage the technology of the future but target today's mobile, desktop and browser environments.

Two-Way Databinding

- Powerful two-way binding to any object.
- Uses adaptive techniques to select the most efficient way to observe each property in your model
- Automatically sync UI with best-in-class performance.

Routing & UI Composition

- Data-driven UI composition
- Advanced client-side router:
  - pluggable pipeline
  - dynamic route patterns
  - child routers
  - asynchronous screen activation

Broad Language Support

- Use ES5, ES 2015, ES 2016 and **TypeScript**.

- Aurelia's APIs were designed to be consumed these languages

Modern Architecture

- Not a monolithic framework
- Composed of smaller, focused modules.
- Use them together as a full-featured framework or pick and choose to build a custom solution.

Extensible HTML

- HTML compiler supports:
  - creation of custom HTML elements
  - addition of custom attributes to existing elements
  - controls template generation
- Supports dynamic loading, databinding and high-performance batched rendering.

MV* with Conventions

- Fully integrated MV* architecture
- Leverages conventions to facilitate seamless app construction
- Plug in conventions to override framework approach

Testable

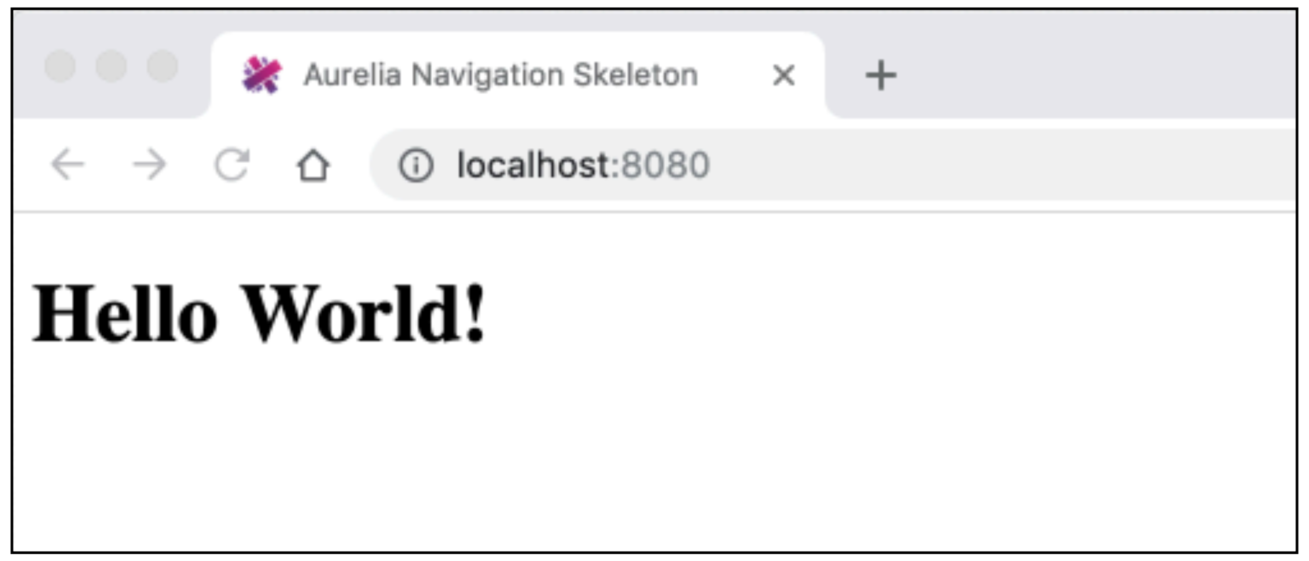
- Combines Javascript/Typescript modules with a Dependency Injection Container,
- Facilitates creation of cohesive, minimally coupled code
- Important for testing

Aurelia Hello World

- Install **yarn**
- Install **aurelia-cli** (command line interface)
- Run the cli
- Launch the generated application
- Browse to https://localhost:8080

```
au new donation-client
```

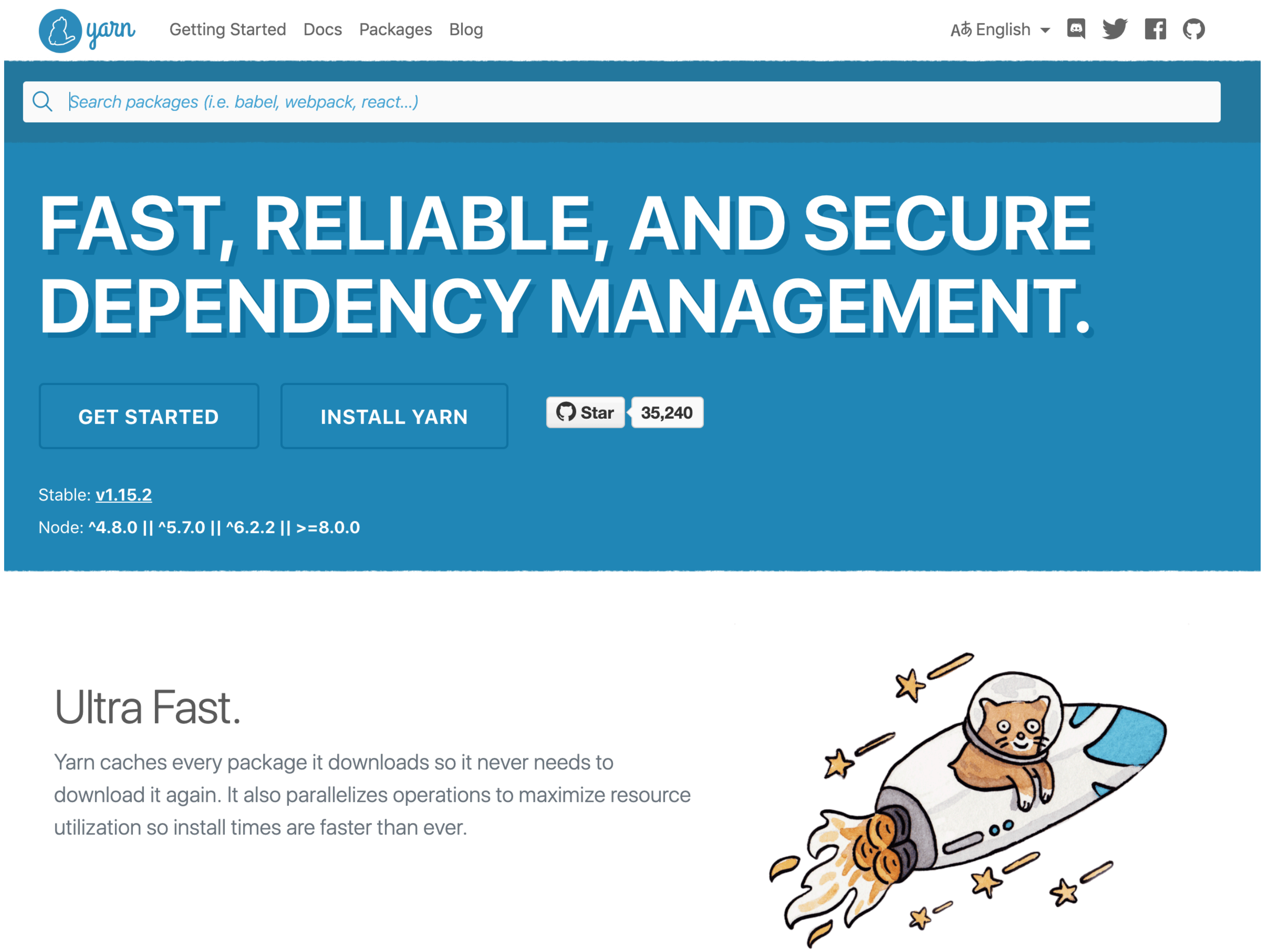
```
au run --watch
```



Hello World!

Yarn

- Alternative to npm
- Faster and more secure



Getting Started Docs Packages Blog

English

Search packages (i.e. babel, webpack, react...)

FAST, RELIABLE, AND SECURE DEPENDENCY MANAGEMENT.

GET STARTED INSTALL YARN Star 35,240

Stable: v1.15.2

Node: ^4.8.0 || ^5.7.0 || ^6.2.2 || >=8.0.0

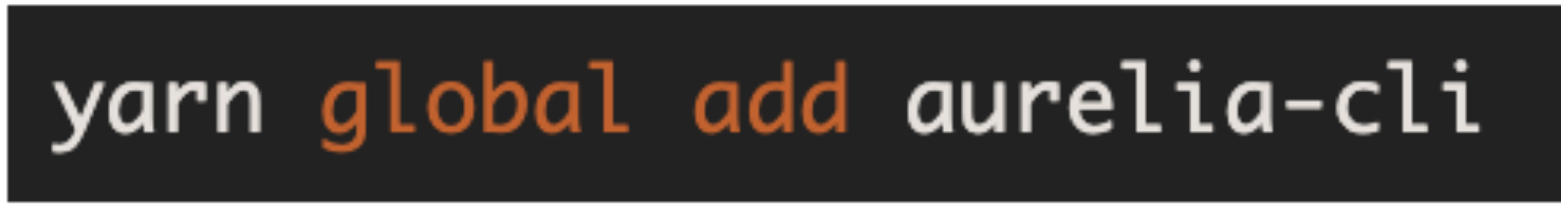
Ultra Fast.

Yarn caches every package it downloads so it never needs to download it again. It also parallelizes operations to maximize resource utilization so install times are faster than ever.

Aurelia Command Line Interface

- The Aurelia CLI is the official command line tool for Aurelia. It can be used to create new projects, scaffold components and to bundle your application for release. It is the best way to get started on a new Aurelia project.

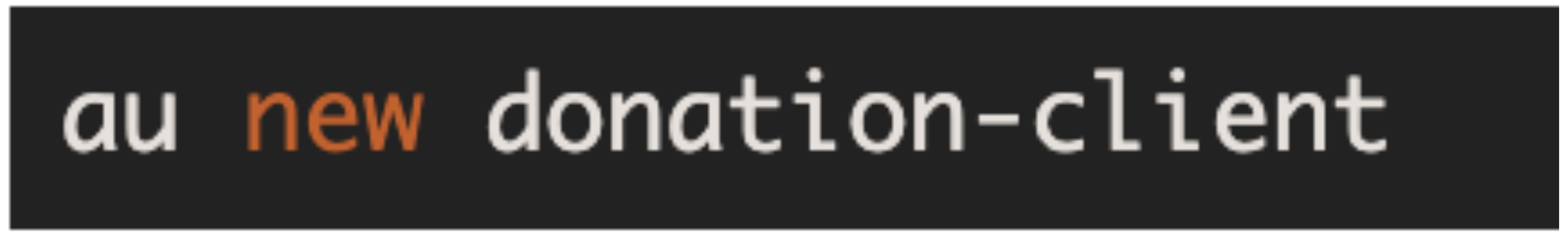
- To install:



```
yarn global add aurelia-cli
```

Creating A New Aurelia Project

- To create a project:



```
au new donation-client
```

- Select ‘TypeScript’ as the default language

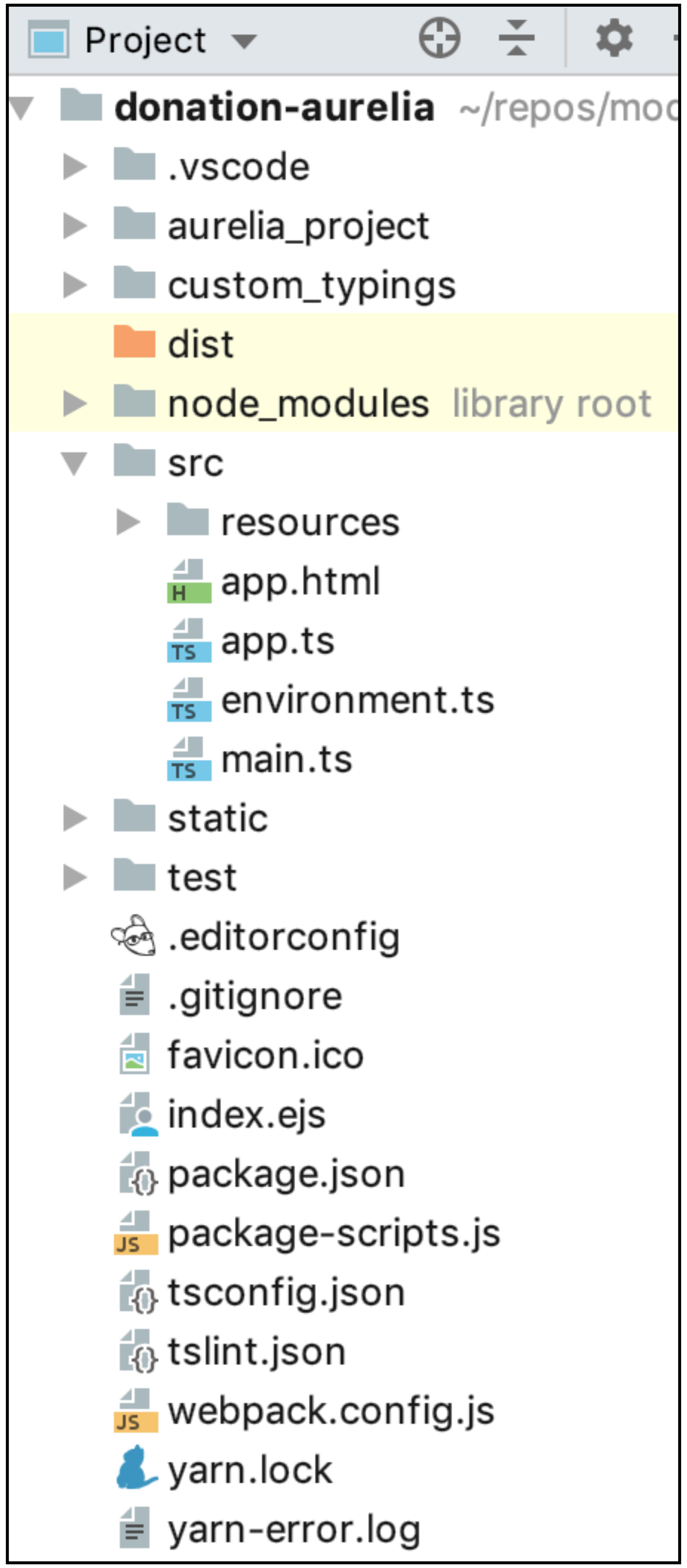
- Select ‘Yarn’ as the default package installer

- Open project in Webstorm

- To launch Enter

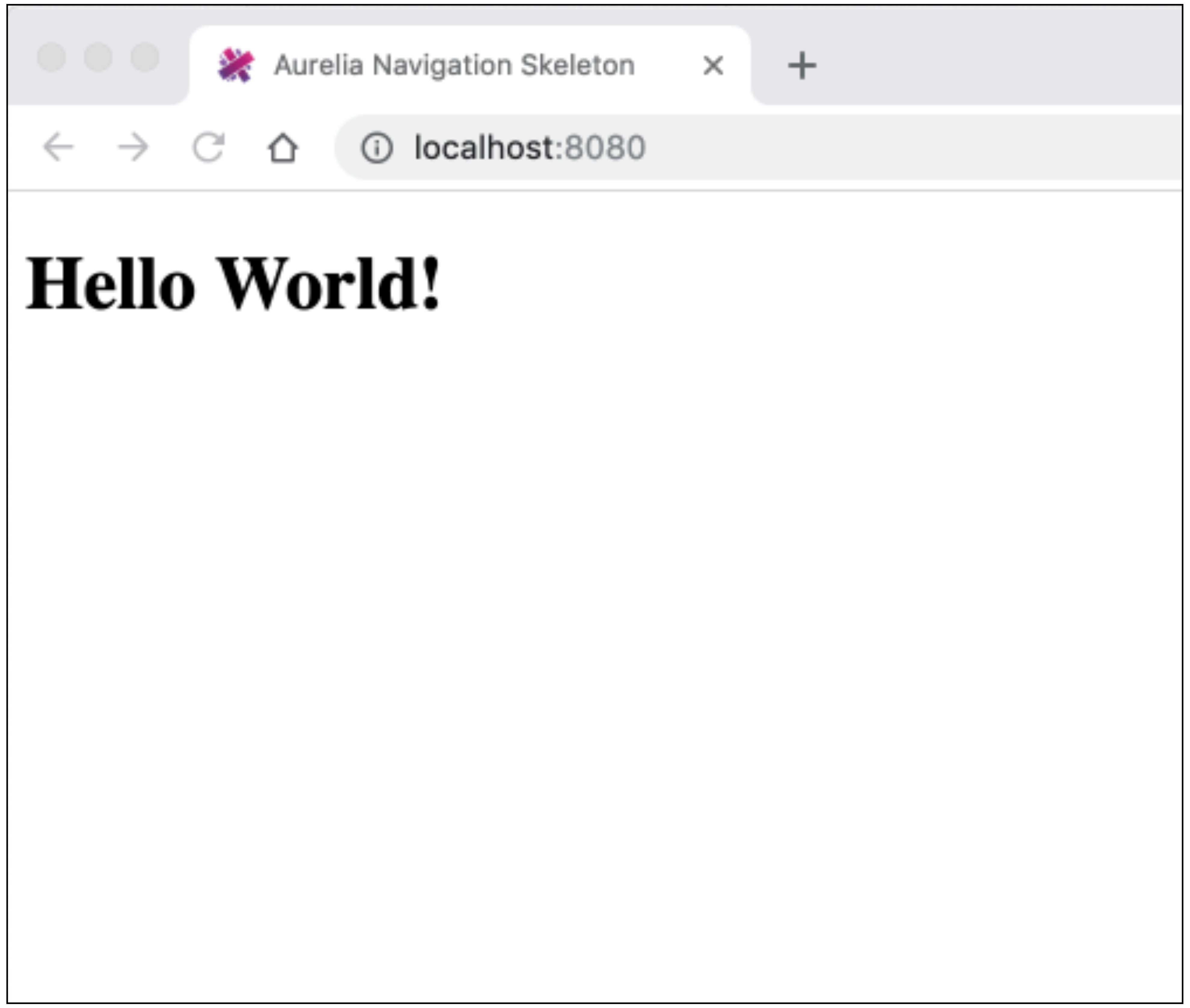
```
au run --watch
```

http://localhost:8080/



Project

donation-aurelia ~/repos/moc
- .vscode
- aurelia_project
- custom_typings
- dist
- node_modules library root
- src
  - resources
  - app.html
  - app.ts
  - environment.ts
  - main.ts
- static
- test
- .editorconfig
- .gitignore
- favicon.ico
- index.ejs
- package.json
- package-scripts.js
- tsconfig.json
- tslint.json
- webpack.config.js
- yarn.lock
- yarn-error.log



Aurelia Navigation Skeleton

localhost:8080

Hello World!