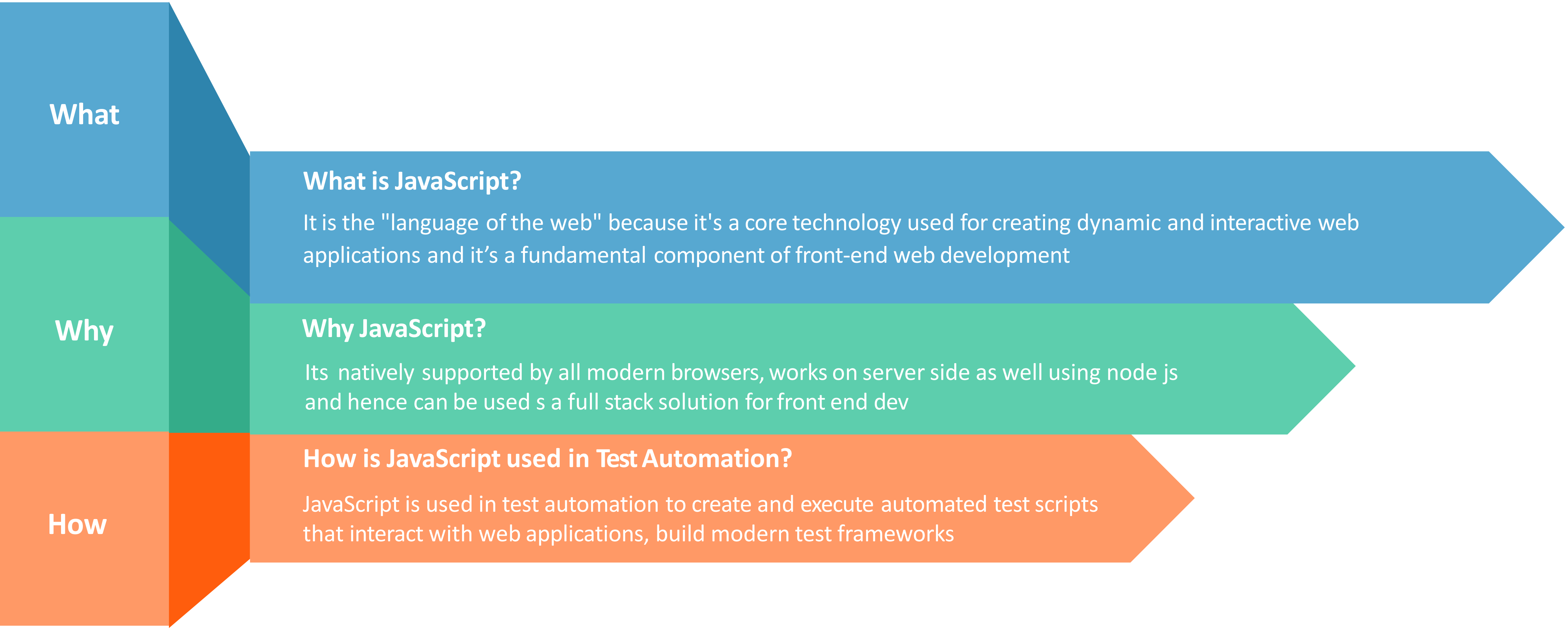
JavaScript Introduction

What, Why & How?

Testleaf
Always Ahead

The Golden Circle



What

Why

How

What is JavaScript?

It is the "language of the web" because it's a core technology used for creating dynamic and interactive web applications and it’s a fundamental component of front-end web development

Why JavaScript?

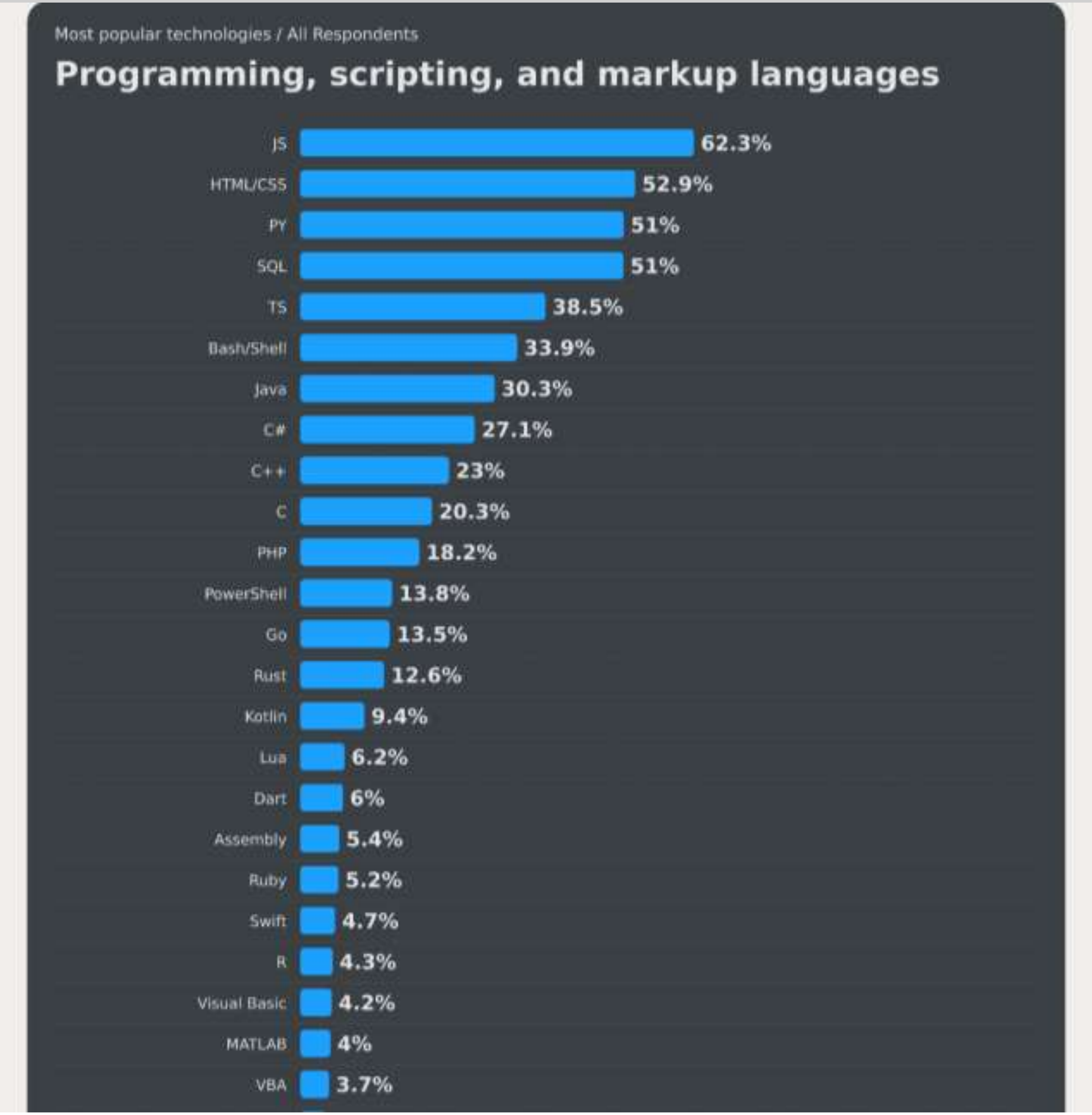
Its natively supported by all modern browsers, works on server side as well using node js and hence can be used s a full stack solution for front end dev

How is JavaScript used in Test Automation?

JavaScript is used in test automation to create and execute automated test scripts that interact with web applications, build modern test frameworks

2024 – Language Forecast

stackoverflow



Most popular technologies / All Respondents

Programming, scripting, and markup languages

How JavaScript Emerged & Evolved?

Brendan Eich

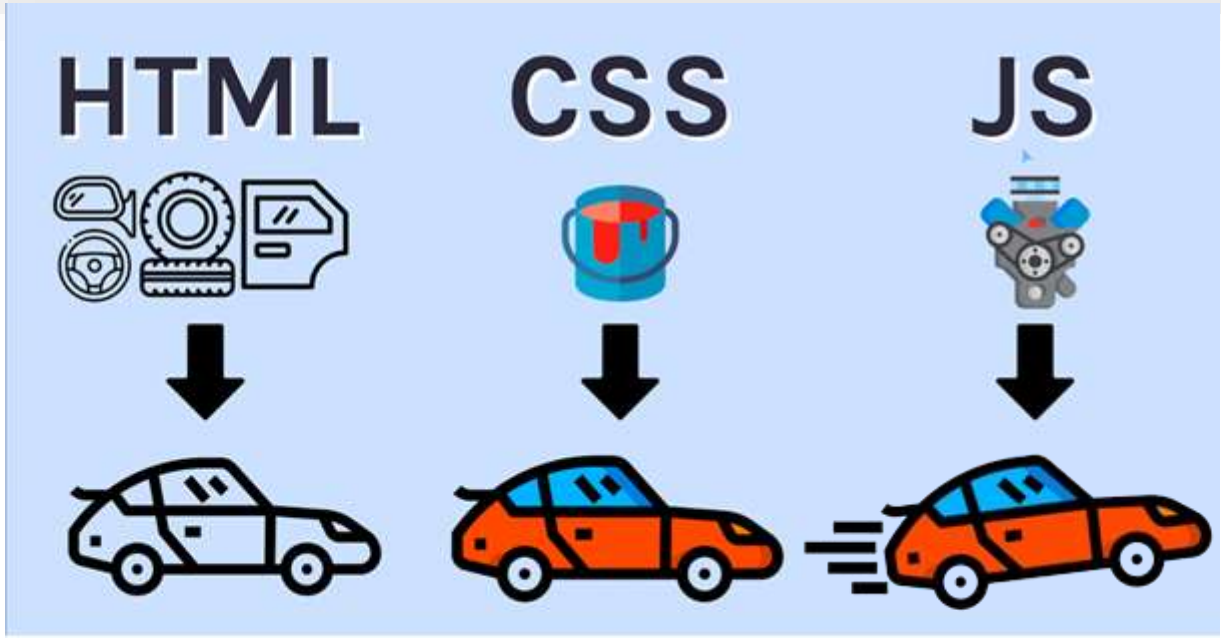
- ❑ **Origin (1995):**
  JavaScript was created by **Brendan Eich** at Netscape.
  Initially, it was called **Mocha**, then **LiveScript**, and finally **JavaScript** to align with Java's popularity at the time, even though it is not directly related to Java.

- ❑ **Purpose:**
  JavaScript was designed to add interactivity to static HTML pages, making web applications dynamic and engaging.

- ❑ **Standardization (1997):**
  To ensure compatibility across browsers, JavaScript was submitted to **ECMA** (European Computer Manufacturers Association) **International**, resulting in the first standard version, **ECMAScript 1**.

- ❑ **Modern JavaScript:**
  JavaScript has evolved significantly, with major updates like ES6 (2015) introducing let, const, classes, arrow functions, and modules, enhancing its power for front-end and back-end development.
  Async/Await, introduced in ES8 (2017), revolutionized how JavaScript handles asynchronous tasks,

From Client side validation → Full-Stack



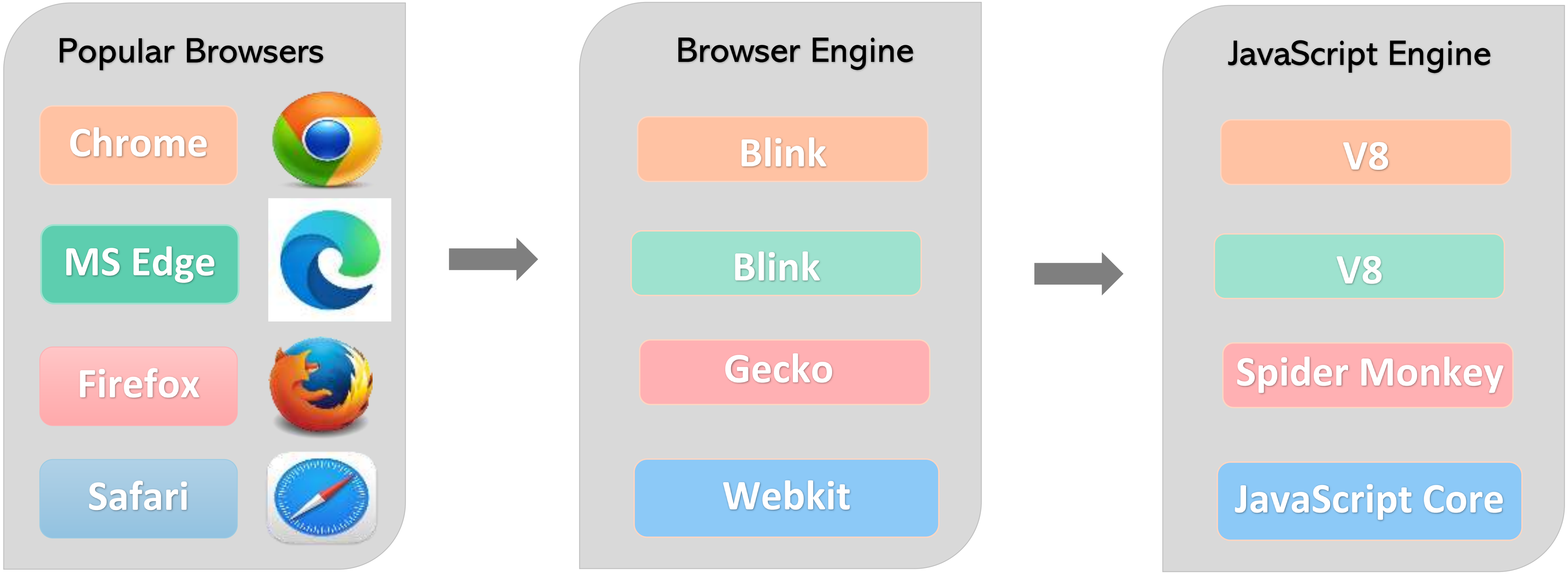
❑ **Early Days (1995):**
JavaScript was initially created to handle **simple client-side validation**, such as checking form inputs before submission.

❑ **Enhanced Browser Support (1997-2005):**
As browsers advanced, JavaScript gained more features and became central to creating interactive web pages.

❑ **AJAX Revolution (2005):**
The introduction of **AJAX (Asynchronous JavaScript and XML)** allowed JavaScript to fetch data from the server without reloading the page, revolutionizing web applications.

❑ **Node.js Introduction (2009):**
Ryan Dahl's introduction of Node.js enabled JavaScript to run on the server side, transforming it into a full-stack solution for building scalable, high-performance applications.

Popular Browsers and their Javascript Engines



Popular Browsers	Browser Engine	JavaScript Engine
Chrome	Blink	V8
MS Edge	Blink	V8
Firefox	Gecko	Spider Monkey
Safari	Webkit	JavaScript Core

Testleaf
Always Ahead

Foundation of JavaScript :

Asynchronous Programming

Asynchronous programming lets tasks run in the background without blocking the main thread, allowing the program to keep executing other tasks. *In JavaScript, it uses callbacks, Promises, and async/await to manage these operations efficiently.*

Event-Driven Architecture

Event-driven architecture designs programs to respond to events, such as clicks, keypresses, or messages, triggering specific actions. *In JavaScript, events emit actions, and event listeners (callback functions) handle them.*

```
document.getElementById("myButton").addEventListener("click", function() {
  alert("Button clicked!");
});
```

✅ Here, JavaScript waits for the "click" event on myButton and runs the function only when the button is clicked.

Single-Threaded with Non-Blocking I/O

JavaScript is single-threaded, with Non-blocking I/O allows input/output operations (e.g., file reads, network requests) to happen without pausing other tasks. *In JavaScript, the program continues execution, and results are processed later using callbacks, Promises, or the event loop.*

Real life analogy on JavaScript

The oven cooks the food while the chef prepares other dishes (**I/O handled without blocking** other operations).

Waiter reacts to customers raising their hands or signaling for service (**events**).

Waiter continues serving other customers while waiting for the food to cook (tasks handled **asynchronously**).