# Browser Runtime Programming Using TypeScript

## File API, IndexedDB, GeoLocation, Beacon, HTTP/x, Events, CORS, Fetch, Formats and Device Info

The programming environment inside the modern web browser has significantly matured and now offers a rich and diverse range of capabilities, some at the UI level and some at the underlying runtime level. Many of the newer features are currently not being fully exploited by web developers, who focus exclusively on the UI. This course aims to change that by exploring in depth the non-UI aspects of browser programming, using the TypeScript language for all demos and lab exercises.

The runtime programming APIs in a modern browser now rival what modern OSes offer. A web app has many natural advantages over mobile apps (here's four: the power of the URI, works everywhere, cloud-friendly, immediate app updates). In areas such as sandboxed file access, networking, data formats and device info, a modern browser offers the web application developer a comprehensive selection of functionality that, when used correctly, can easily compete with what is available for native mobile apps.

IMPORTANT: This course does not cover parallel (web worker) & asynchronous programming – we offer a separate full course that covers these topics in detail.

### Target Audience

Web developers wishing to fully leverage the runtime (non-UI) capabilities of modern web browsers.

### Prerequisites

Good experience of web development, including HTML 5.3.

Knowledge of the TypeScript programming language.

### *Contents of One-Day Training Course*

#### Modern Web Platform

Tour of the modern web platform (what we already know / what is less familiar)
Major features and which browsers implement them (caniuse.com and iwanttouse.com websites)
Dynamically detecting in code available browser capabilities and optimizing app

#### File API

File Sandboxing
Accessing and updating files
File metadata and raw blob content
File[Reader|Writer|system] API

#### Indexed DB

A B-tree like persistence mechanism with powerful indexing that can serve as basis for in-browser client database
Comparison with other storage options

#### Text Handling And JSON

Handling various text encodings
TextEncoder and TextDecoder
JSON parsing and generation
Use of JSON in the browser

#### GeoLocation

"This specification defines an API that provides scripted access to geographical location information associated with the hosting device."

#### Beacon

Sending data asynchronously to server (e.g after page close)
navigator.sendBeacon
Setting priority

#### HTTP/2 & HTTP/3

Latest generation of HTTP protocol
Binary, multiplexed, full duplex, priority
Server push
New app architectural possibilities

#### Server Sent Events

How it works
EventSource API
Event streams
Event handlers

#### CORS

Cross-Origin Resource Sharing
Role of origin in HTTP protocol
cross-origin access
CORS preflight request and CORS request

#### Fetch

A significantly improved replacement for XmlHttpRequest
"The Fetch standard defines requests, responses, and the process that binds them: fetching." (WHATWG)

#### Data Formats

Brotli Compressed Data Format
Data URIs using base64 encoding
data-* attributes

#### Device Info

Accessing device details
Vibration API
Battery status

#### Project

Bringing together the ideas covered in this course to design and build a specialist runtime engine to execute in a browser