| **Topic** | **Basic** | | **Advanced** | **Expert** |
| --- | --- | --- | --- | --- |
| **JavaScript in WebBrowser** | | | | |
| **DOM** | 1. Understand the **HTML document structure**, the **tree** concept; 2. Be aware of how **whitespaces** are handled in DOM; 3. Be aware of HTML entities; 4. Know the various DOM elements **content types**, text nodes; 5. Know how to **traverse the DOM**; 6. Know how to **retrieve elements** from DOM, selectors; 7. Know how to **alter DOM** elements; 8. Know how to **create DOM** elements; 9. Know how to **modify attributes**, **styles**, **CSS**; 10. Know how to work with **positioning** & **animation**; 11. Be aware of **content editability** feature; 12. Know the **form elements**, purpose of each; | | 1. Know what is a **Document Fragment** and its purpose; 2. Understand how to work with **geometry** of a document & **scrolling**; 3. Understand the concepts behind **text selection API**; | 1. Know what is Shadow DOM and its purpose; 2. Know the **Intersection Observer API**; |
| **Event handling** | 1. Know the browser **event types** (basic):    1. Network events    2. Focus events    3. Form events    4. Keyboard events    5. Mouse events    6. Drag & Drop events    7. Document load 2. Know the ways of **events registering**; 3. Know the **events firing** mechanism:    1. Argument    2. Context    3. Returning value of event handlers    4. Calling precedence    5. Event propagation (stopPropagation / stopImmediatePropagation)    6. Event canceling    7. Default actions | | 1. Know the **throttling** technique and the situations when it is useful; 2. Know the **performance aspects of events** handling; 3. Ability to describe the **event delegation** concept. 4. Know the difference between **target & currentTarget**. 5. Know the **3rd parameter of addEventListener**:    1. capture    2. once    3. passive | 1. Know the browser **event types** (**advanced**):    1. Progress events    2. Resource events    3. WebSocket events    4. CSS animation events    5. CSS Transition events    6. Printing events    7. Text composition events    8. View events    9. Session history events    10. Clipboard events    11. Media events    12. Storage events    13. Value change events    14. WebVR events |
| **Networking** | 1. Know how **XMLHttpRequest** / **FormData** work 2. Know how **fetch** works; 3. Know how **WebSocket** works; 4. Ability to describe in which situation a certain networking technique should be used; | | 1. Know the patterns of **network traffic minimization**; 2. Be aware of **offline apps techniques**; |  |
| **Client Side data storage** | 1. Know how to work with **localStorage**, **sessionStorage** (Storage API), their limitations and purpose; 2. Know what are the **cookies**, their purpose, common use cases and how to work with them; | | \_empty\_ | 1. Know the techniques of **big data storage** on the client side. Common use cases; |
| **Web APIs / interfaces** | 1. Know the **browser’s APIs/interfaces** (basic):    1. Window object    2. Timers    3. History    4. Location    5. Navigator    6. Screen    7. Working with windows & frames    8. File API | | \_empty\_ | 1. Know the **browser’s APIs/interfaces** (advanced):    1. Threads, web workers:       1. SharedWorker.       2. ServiceWorker.       3. ChromeWorker.       4. AudioWorker.    2. Dialogs    3. Canvas    4. Audio/Video    5. SVG    6. Geopositioning    7. Client-side DBs    8. Device/Screen orientations    9. Ambient Light Sensor    10. Proximity API    11. Vibration API    12. Pointer Lock API    13. Web RTC    14. Payment request |
| **Misc** | 1. Know how to connect JSinto HTML website; 2. Understand what is line terminations (**line endings**) LF/CRLF and what problems can occur because of negligence; | | 1. Know the tips for **fast-loading** of HTML pages:    1. Reduce page weight.    2. Minimize the number of files.    3. Use a CDN.    4. Reduce domain lookups.    5. Cache reused content.    6. Page components ordering optimally.    7. Content chunking.    8. Images minimizing & compression.    9. Prevent reflow via size specification for images & tables.    10. Lazy loading for images.    11. Async & defer usage. | 1. Understand the **PWA** concept; |
| **V8** | \_empty\_ | | \_empty\_ | \_empty\_ |