



JavaScript

# Why learn pure JavaScript?

- “Learning JavaScript is like learning how to build a house from scratch!”

Vanilla JavaScript μας επιτρέπει:

- Να μαθαίνουμε με ευκολία νέα JS frameworks:
- Πολλές φορές πρόκειται να χρησιμοποιήσουμε pure JS στον κώδικά μας, οπότε πρέπει να είμαστε σε θέση να το κάνουμε
- Πρέπει να καταλαβαίνουμε τι γράφουμε και γιατί
- Να μπορούμε να κάνουμε παραμετροποιήσεις ανεξαρτήτως framework/βιβλιοθήκης που χρησιμοποιούμε

# Why learn JavaScript?

Vanilla JavaScript μας επιτρέπει:

- Να έχουμε καλύτερη ποιότητα στον κώδικά μας
- Να διορθώνουμε εύκολα πιθανά προβλήματα στον κώδικά μας
- Δεν θέλουμε να δεσμευόμαστε με frameworks/βιβλιοθήκες
- Θέλουμε να μπορούμε να μαθαίνουμε εύκολα νέα frameworks/βιβλιοθήκες

# To begin with...

- Θα χρησιμοποιήσουμε απλές HTML pages για να εξοικειωθούμε με την vanilla JavaScript

# HTML5 introduction

- Hyper Text Markup Language (HTML) ναι η στανταρ γλώσσα που χρησιμοποιείται για την ανάπτυξη ιστοσελίδων.
- Η HTML περιγράφει τη δομή μιας ιστοσελίδας και αποτελείται από μια σειρά στοιχείων (*elements*)
- Τα στοιχεία καθορίζουν το περιεχόμενο που εμφανίζεται στο πρόγραμμα περιήγησης (π.χ. εικόνα, σύνδεσμος, video κ.λπ.)

# HTML Elements

- Ένα HTML **element** είναι ένα μεμονωμένο στοιχείο ενός εγγράφου HTML.
- Αντιπροσωπεύει σημασιολογία ή νόημα
  - i.e title element: αντιπροσωπεύει τον τίτλο του εγγράφου.
- Τα περισσότερα HTML elements ξεκινούν με ένα *opening tag* και τελειώνουν με ένα closing tag. Το περιεχόμενο τοποθετείται ανάμεσα στα 2 tags.
- Τα στοιχεία (elements) μπορούν επίσης να περιέχουν χαρακτηριστικά (attributes) που καθορίζουν τις πρόσθετες ιδιότητές τους. Για παράδειγμα, μια παράγραφος, η οποία αντιπροσωπεύεται από το στοιχείο p, θα γραφόταν ως:

# HTML Elements

- i.e a paragraph with attribute would be written:

```
<p class="sample">This is a paragraph with attribute </p>
```

# Elements example

- Below are some examples of elements in HTML

```
<h1> My First Heading </h1>
```

```
<p> My first paragraph </p>
```

- Empty elements: do not have contents and end tag  
i.e. <br>



# HTML Basics 1/2

- HTML documents: πρέπει να ξεκινούν με μια δήλωση του τύπου εγγράφου (type declaration): **<!DOCTYPE html>**: βοηθά τους browsers να εμφανίζουν σωστά τις ιστοσελίδες
- Το HTML document είναι δομημένο χρησιμοποιώντας: `<html>`  
`</html>`
- Το ορατό τμήμα του εγγράφου HTML βρίσκεται ανάμεσα στα tags: `<body>` `</body>`

# HTML Basics 2/2

- An example of HTML structure for a simple "Hello World" message:

```
<!DOCTYPE html>  
<html>  
  <body>  
    <p>Hello World</p>  
  </body>  
</html>
```

# Nested HTML Elements Example

```
<!DOCTYPE html>
```

```
<html>
```

```
  <head>
```

```
    <title>My Nested Elements </title>
```

```
  </head>
```

```
  <body>
```

```
    <h1>Hello <b>World</b> </h1>
```

```
    <p>Do not be afraid to <u>underline</u> what's important</p>
```

```
  </body>
```

```
</html>
```

# Basic HTML Elements

Tag	Description
<a href="#"><u>&lt;!DOCTYPE&gt;</u></a>	Defines the document type
<a href="#"><u>&lt;html&gt;</u></a>	Defines an HTML document
<a href="#"><u>&lt;head&gt;</u></a>	Contains information regarding the document
<a href="#"><u>&lt;title&gt;</u></a>	Title of the document
<a href="#"><u>&lt;body&gt;</u></a>	Defines the document's body
<a href="#"><u>&lt;h1&gt; to &lt;h6&gt;</u></a>	Defines HTML headings
<a href="#"><u>&lt;p&gt;</u></a>	Paragraph

# Basic HTML Elements

Tag	Description
<a href="#"><u>&lt;br&gt;</u></a>	Inserts single line break
<a href="#"><u>&lt;a&gt;</u></a>	Hyperlink
<a href="#"><u>&lt;button&gt;</u></a>	A clickable button
<a href="#"><u>&lt;img&gt;</u></a>	Defines an image
<a href="#"><u>&lt;map&gt;</u></a>	Defines a client-side image map
<a href="#"><u>&lt;link&gt;</u></a>	Link to an external resource (most used to link to style sheets)
<a href="#"><u>&lt;div&gt;</u></a>	Defines a section in a document

# Basic HTML Elements

Tag	Description
<a href="#"><u>&lt;header&gt;</u></a>	Defines a header for a document or section
<a href="#"><u>&lt;footer&gt;</u></a>	Defines a footer for a document or section
<a href="#"><u>&lt;script&gt;</u></a>	Defines a client-side script
<a href="#"><u>&lt;span&gt;</u></a>	A section in a document

- Ας ανακεφαλαιώσουμε εν συντομία τι συμβαίνει όταν φορτώνεται μια ιστοσελίδα σε ένα πρόγραμμα περιήγησης
- Όταν φορτώνετε μια ιστοσελίδα στον **browser** (HTML, CSS και JavaScript) , εκτελείτε τον κώδικα της σελίδας μέσα σε ένα **περιβάλλον εκτέλεσης** (την καρτέλα του προγράμματος περιήγησης).
- Μπορείτε να το φανταστείτε σαν ένα εργοστάσιο που παίρνει πρώτες ύλες (τον κώδικα) και βγάζει ένα προϊόν (την ιστοσελίδα).

- Μια πολύ κοινή χρήση της JavaScript είναι η δυναμική τροποποίηση της HTML και της CSS για να αλλάξει το user interface
- HTML allows us to use Javascript using:
  - Internal JavaScript: the `<script>` Tag
  - an External JavaScript

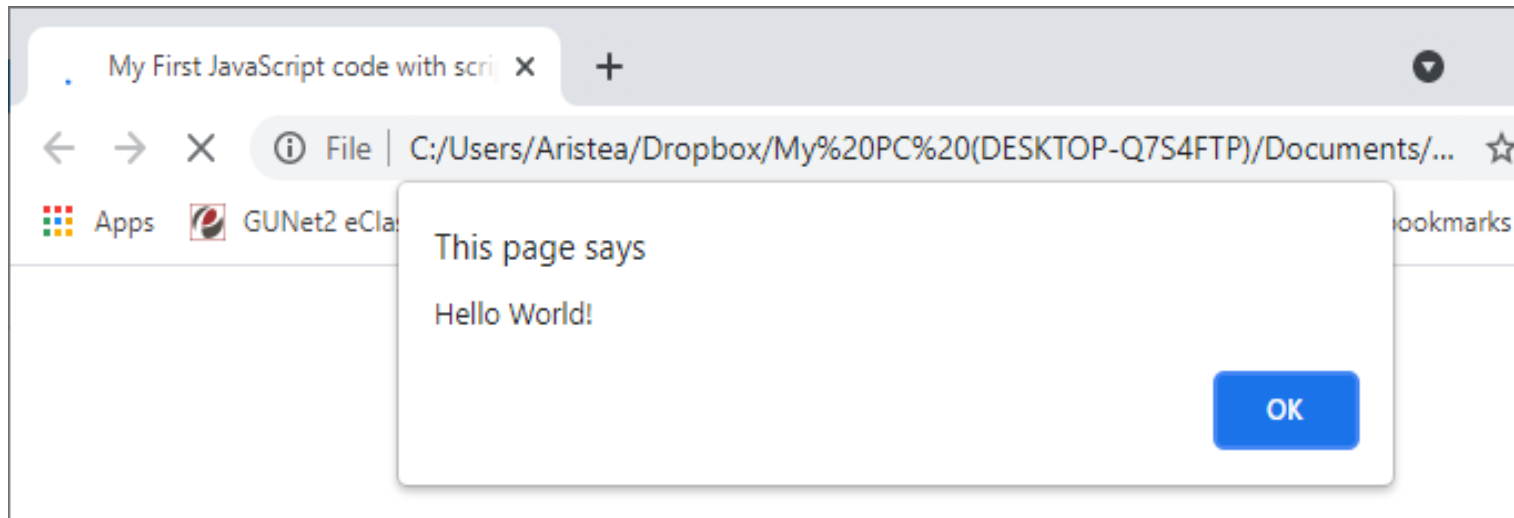


# <script> Tag

- Η ετικέτα <script> χρησιμοποιείται κατά την τοποθέτηση του κώδικα JavaScript στο έγγραφο HTML.
- <script> JavaScript code </script>
- Σημείωση: στην HTML5 το <script type="text/javascript"> δεν απαιτείται
- Η JavaScript είναι η default scripting language στην HTML

# <script> Tag Example

```
1 <html>
2 <head>
3   <title>My First JavaScript code with script tag!!!</title>
4   <script>
5     alert("Hello World!");
6   </script>
7 </head>
8 <body>
9 </body>
10 </html>
```



# Internal JS

- Εάν έχετε μόνο λίγες γραμμές κώδικα που είναι συγκεκριμένος για μια συγκεκριμένη ιστοσελίδα, τότε είναι καλύτερο να διατηρήσετε τον κώδικα JavaScript εσωτερικά μέσα στο HTML αρχείο.
- Από την άλλη πλευρά, εάν ο κώδικας JavaScript χρησιμοποιείται σε πολλά αρχεία, τότε θα πρέπει να εξετάσετε το ενδεχόμενο να διατηρήσετε τον κώδικα σε ξεχωριστό αρχείο.

# External JavaScript

- common practice-> place scripts in external files
- JavaScript files : .js extension  
`<script src="script.js" ></script>`
- External scripts είναι μια καλή επιλογή όταν θέλουμε να:
  - έχουμε επαναχρησιμοποιήσιμο κώδικα
  - Να έχουμε χωριστά τα HTML , CSS & JS αρχεία
  - Βελτιώνει τη συντήρηση και την κατανόηση των scripts

Αποτελεί γενικά μια καλή προσέγγιση όσον αφορά την οργάνωση του κώδικα και την επαναχρησιμοποίησή του σε πολλά αρχεία HTML. Επιπλέον, γίνεται ευκολότερη και η ανάγνωση της HTML

# Js in HTML document

- Περισσότερα από ένα script μπορούν να εισαχθούν σέ ένα HTML document
- Το `<script>` tag μπορεί να εισαχθεί :
  - στο `<body>`
  - στο `<head>`
  - Και στα 2

# Script loading strategies

- Υπάρχει μια σειρά ζητημάτων που σχετίζονται με τη φόρτωση των scripts την κατάλληλη στιγμή.
- Ένα κοινό πρόβλημα είναι ότι αν χρησιμοποιείτε JavaScript για να χειριστείτε elements στη σελίδα ο κώδικάς σας δεν θα λειτουργήσει εάν ο κώδικας JavaScript φορτωθεί και εκτελεστεί πριν από το HTML element
- Η παραδοσιακή λύση στο πρόβλημα αυτό είναι να βάζουμε το script element ακριβώς πριν το `</body>` tag.

Αλλά αυτή η λύση απέχει πολύ από το να είναι τέλεια. Για παράδειγμα, ο browser “βλέπει” το script (και μπορεί να ξεκινήσει τη λήψη του) μόνο αφού κατεβάσει το πλήρες HTML document. Για μεγάλα HTML αρχεία, αυτό μπορεί να είναι μια αισθητή καθυστέρηση.

// μπορεί μια τέτοια καθυστέρηση να γίνει αντιληπτή σε άτομα που έχουν χαμηλές ταχύτητες Διαδικτύου

# Script loading strategies

- Υπάρχουν δύο σύγχρονες δυνατότητες που μπορούμε να χρησιμοποιήσουμε σε external files για να παρακάμψουμε το παραπάνω πρόβλημα— **async** και **defer**
- **defer** :
- `<script defer src="myscript.js"></script>`
- Τα scripts που φορτώνονται χρησιμοποιώντας το defer θα φορτωθούν με τη σειρά που εμφανίζονται στη σελίδα. Δεν θα εκτελεστούν όμως έως ότου φορτωθεί όλο το περιεχόμενο της σελίδας.

# Script loading strategies

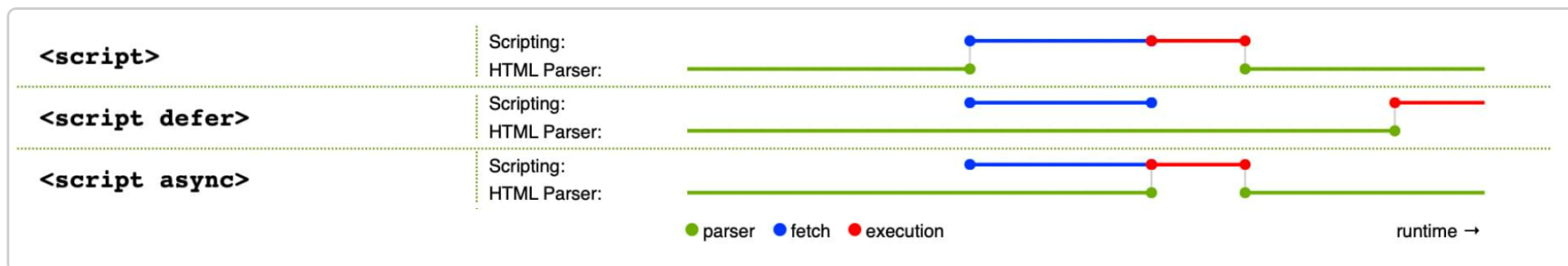
- **defer :**
- **<script defer src="script1.js"></script> (1)**
- **<script defer src="script2.js"></script> (2)**
- **<script defer src="script3.js"></script> (3)**



# Script loading strategies

- **Async:**
- `<script async src="myscript.js"></script>`
- Τα scripts που φορτώνονται χρησιμοποιώντας `async` θα κάνουν download το script χωρίς να μπλοκάρουν την υπόλοιπη σελίδα. Μόλις ολοκληρωθεί η λήψη, θα εκτελεστεί το script.
- Βέβαια, σε περίπτωση με περισσότερα από ένα scripts στη σελίδα, δεν έχουμε **κάποια εγγύηση ότι τα θα εκτελεστούν με συγκεκριμένη σειρά**. Είναι καλύτερο να χρησιμοποιείτε το `async` όταν τα scripts στη σελίδα εκτελούνται ανεξάρτητα .

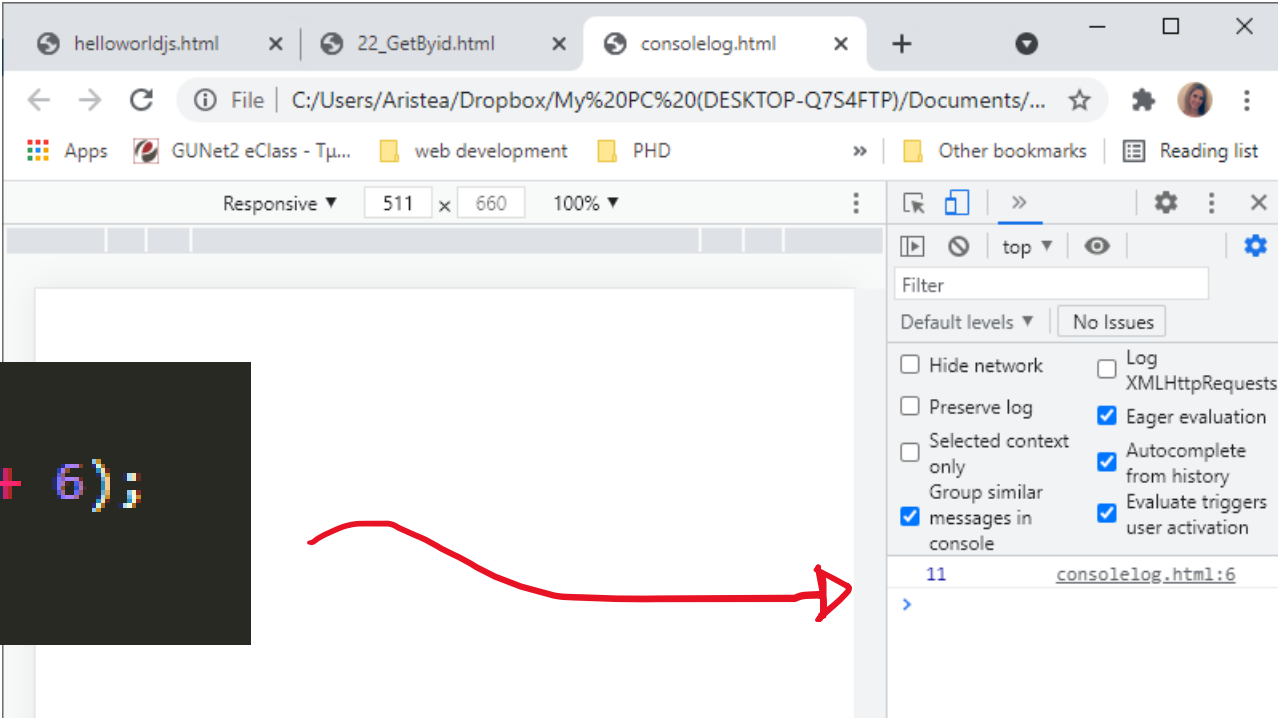
- **async & defer** : instruct the browser to download the script(s) in a **separate thread**, while the rest of the page is downloading (page loading is not blocked)
- scripts with an **async attribute** will execute as soon as the download is complete. This blocks the page and does not guarantee any specific execution order.
- scripts with a **defer attribute** will load in the order they are in and will only execute once everything has finished loading.



Link: <https://html.spec.whatwg.org/images/asyncdefer.svg>

# Use JS to display data

- `console.log()`: write into the browser console



The image shows a browser window with three tabs: `helloworldjs.html`, `22_GetByid.html`, and `consolelog.html`. The address bar shows the file path: `C:/Users/Aristea/Dropbox/My%20PC%20(DESKTOP-Q7S4FTP)/Documents/...`. The browser interface includes a toolbar with navigation buttons, a search bar, and a bookmark bar. The main content area is currently blank. A red arrow points from a code block on the left to the browser's developer console on the right.

```
<script>  
console.log(5 + 6);  
</script>
```

The developer console on the right shows a single log entry at line 11 of `consolelog.html`. The console is currently empty, with a blue prompt character `>` visible below the log entry.

# JS Comments

- Comments are useful in Javascript code in order to make it more readable or just to prevent execution while testing

- Single line comments: //

- Multiple line comments

```
/*  
These are  
Comments of course!  
*/
```



*To be continued...*