CODEFIRST: Professionals

Web Fundamentals - Session 4

This week's agenda

The agenda for today is to:

1. Revise the key topics we've already covered so far
2. Introduce you to: Web Frameworks & some other basic web development concepts
3. Start your project website if you haven't already

Topics we've covered so far

Session 1: Getting going + HTML

- HTML syntax

Session 2: CSS

- Tags, Selectors and Attributes, Stylesheets

Session 3: GitHub & the Command Line

- How to use GitHub - what is version control? (basic concepts, commits, pulls, forks, etc)
- GitHub Pages (https://pages.github.com/), hosting your website
- Introduction to the Command Line

Task 1 : Take a quick look through the session notes from the last three sessions with your partner. If you're unclear on any of the concepts work through them with your partner and your instructor.

Introduction to Web Frameworks

A web application framework is a **type of software framework** designed to **support development** of dynamic websites, web applications, web services and of web resources.

A **software framework**, in computer programming, is an abstraction in which **common code** providing **generic functionality** can be selectively *overridden* or *specialized* by user code providing specific functionality.

Frameworks are a **special case** of software libraries in that they are **reusable** abstractions of code wrapped in a well-defined Application Program Interface (API),

yet they contain some key distinguishing features that separate them from normal libraries.

Software frameworks have these distinguishing features that separate them from libraries or normal user applications:

1. **inversion of control** - In a framework, unlike in libraries or normal user applications, the overall program's flow of control is not dictated by the caller, but by the framework.[1]
2. **default behavior** - A framework has a default behavior. This default behavior must actually be some useful behavior and not a series of no-ops.
3. **extensibility** - A framework can be extended by the user usually by selective overriding or specialized by user code providing specific functionality.
4. **non-modifiable framework code** - The framework code, in general, is not allowed to be modified. Users can extend the framework, but not modify its code.

By using frameworks, we benefit from **peer-reviewed**, **tested** and **pre-written functionalities** that save us time and allow us to share and contribute to the wider developer community. More here.

https://www.quora.com/What-is-an-API-4

More Basic Development Concepts

1. An **API**, **Application Program Interface**, is an abstract description of how to use an application; it is a set of routines, protocols, and tools for building software applications. The API specifies how software components should interact and APIs are used when programming graphical user interface (GUI) components; allowing you to use code in an already functional application in a stand-alone fashion.
2. A **library** is an implementation of an API; it is a set of functions that a developer can call, usually organised into classes. It contains the compiled code that implements the functions and protocols (maintains usage state).
3. A **toolkit** is a set of libraries (APIs) and services grouped together to provide the developer with a wider range of possible solutions.
    a. For example, the Globus Toolkit provides services (such as File transferring, Job Submission and Scheduling) that a developer can install and start on their servers. They also provide API's to build applications that may use the services deployed in an integrated fashion. For example, the developer may build a

program that uses the Job Submission API to communicate with the Job Submission Service.

Start working on your course competition website!

You've spent 3 weeks learning how to make websites, and now it's time to put those skills to the test! Your task is to make a website of your choosing, in 4 weeks working in pairs.

The criteria we'll be looking at are as follows

- A visually appealing design, including:
    - A contact form (for example name and email)
    - Social buttons
    - As many different HTML elements you can manage
- Interactive elements (like forms) on your website don't need to be functional, but should be present if they need to be for the visual aspect of the design.
- Responsive layouts
    - Viewable on phones, tablets, and computers, with appropriate changes for each.
- A live website
- Good formatting
    - Code split into the appropriate files
    - Files indented properly
- Good organisation
    - Version control using git
    - Sensible git commit messages

So that's it! Other than that, you can make any kind of website you want.

You can see a few examples of what previous students on your course created on our website here: http://www.codefirstgirls.org.uk/course-competition.html

If you're short on ideas, here are some to get you going...

- A personal website
- "How to" website on an area of your expertise

- A survey or poll website.

The websites will be judged by your instructors in the last session. The winning website will receive a prize of £20 Amazon vouchers per team member, and will be featured on the Code First: Girls website!

As always, if you have any questions, don't be afraid to ask, via email, or during the session, and most importantly, have fun!

Task 2 : Make sure you're clear about the course competition requirements. Now's your chance to start your own website!

Last week, your instructor showed you how to put a website up using GitHub Pages, do a recap with your partner, or ask your instructor for a demo again.

Work with your partner and instructor to:

1. Create your project repository on Github
2. Start a new folder on your computer to add your HTML & CSS files. Be sure to link your CSS to your HTML file, see your Session 2 notes and talk to your instructor if you're unclear on how to do this.
3. Start writing your HTML code!