FRONT END – PART 1

MORE ON HTML

Add Semantic Tag to Blog

Now, knowing the importance of DIV and Semantic tags, we will apply it to our blog page as well.

So, divide your page into several sections eg., header, main, etc., whichever apply in your case.

Remember, we use DIV to divide the page into meaningful sections and sometimes to apply some properties.

So, there's nothing wrong and right while dividing your page, just don't add unnecessarily.

Some hints are provided below.

Note: Always apply one

just after tag and then put everything inside it, i.e. header, main and footer.

Hints to add semantic tags -

1. Make header section for the web page.
2. Make the main section for the blog.
3. Make header section for the blog.
4. Make article section for the blog.

The blog page look the same as in the image below -

blog.codingninjas.in

Coding Ninjas Official Blog

Best Coding Practices for Hassle-free Programming

Just like with any other activity, the world of coding is also governed by informal rules. Most of these rules are formulated over decades. Programming languages often remain in usage for longer than their founders expect.

Programmers find innovative ways to recreate these languages for different purposes. Thus, some static do's and don'ts are always of use in the fluid zone called coding. Normally, a great code has three common features: maintainability, dependability, and usability. It should also be adaptable enough to be efficient even in changing environments. However, before coding even begins, a few prerequisites need to be fulfilled in order to provide a solid foundation on which readable code can be written.

PREREQUISITES

This is easily the most important step of all. In absence of these, a code may be inefficient even if it is completed. The prerequisites that need to be in place for an efficient code covers matters like design and architecture. An important question is how development is structured. What is the specific purpose of the software? The problem that the code is meant to solve must be clear to the developer. What is the design of the individual components? Such basic requirements have to be fulfilled when a good code is being written.

Hoare makes a good point when it comes to assessing software design and architecture. Either a code is so simple that there are obviously no errors, or a code is so complex that there are no obvious errors. Software design is also of importance to clients, who can decide if their needs are being adequately addressed. If there are some things that architecture cannot take care of, then those are handled in the software design phase. A detailed design is a good guide for coding.

The next step is to measure existing programming languages against the requirements to decide on an appropriate one. If needed, a mixed routine can be adopted.

Once these are met, it becomes easier to follow some established good coding practices.

- **COMMENTING**

  Commenting is one such practice that is often ignored by programmers, especially for codes that are written by multiple programmers. However, comments reduce the cost of knowledge transfer. Name of the module, purpose of the module, description of the module, original author, modifications and authors who modified code with a description on why it was modified were components of an early commenting practice.

- **GOOD NAMING CONVENTIONS**

  Using good naming conventions is the next step to good coding. Often developers use Y1 or X1 as variables, and then do not replace them with meaningful names. To prevent this confusion and utter waste of time, descriptive names should be used.

- **SIMPLICITY AND COMPACTNESS**

  The key to a good code is also simplicity. To achieve a certain result, developers may use complex logic but this should be minimised while writing a good code. The next developer who uses the code may use it for a different purpose and may not even completely understand or utilise the complex logic. Simple code does not always depend on the length of the code. The code must also be conceptually simple. An added advantage is if the code is also suitably compact. More code can be viewed on a page and programmer keystrokes can be reduced.

- **PORTABILITY**

  Another feature of a good code is portability. If the code contains values referring to 'absolute parameters' such as file names, URLs or IP addresses, it will not run on a host that is of a different design. A good programmer will parametrize these variables and configure them so the application can also be run and maintained in a different environment.

- **TESTING**

  Testing is also a necessary part in the process of coding. Test cases must be planned while the code is being written, and they are developed while an application is coded.

- **DEBUGGING THE CODE**

  While writing a code, it is necessary to check for errors throughout. Though it is fine for smaller programs to be debugged after they have been written, this becomes a bottleneck for longer programmes. The best practice, therefore, is to debug each module as soon as it has been created.

- **DEPLOYMENT**

  Once the code has been written, it has to be deployed. The installation files should be kept to a minimum and updated regularly. Unused code or anything that is not necessary should not be installed.

The best practices for coding are, most of all, designed to reduce risk whenever there are chances of it. For a beginner, a few of these will not be completely applicable. But as a hard-boiled programmer writing numerous codes, all these guidelines will come in handy. Since codes can run up to millions of lines at times, outdated advice like 'keep it simple' or 'less is more' is priceless and should always be remembered.