"Financial Calculators"

Workbook 2's Workshop

What Are Workshops?

During many of the JavaScript coding sessions, you will code a workshop that resembles a 1-1/2 day mini-capstone. Its goal is to reinforce everything you learned about JavaScript during the week in a larger project, while also remembering your HTML, CSS and Bootstrap.

We suggest you create a folder just for these workshops under `LearnToCode` named `Workshops`. Each workshop will be in its own GitHub repository, but the local repository on your machine for each workshop will be stored inside the `Workshops` folder.

While it is important to make these projects attractive, responsive, etc., you don't have to focus quite as much on the UI. Pick simple themes. Don't spend too much time on fancy things like carousels or rounding the edges of images. Make sure you get the logic working and the code follows best practices! If you have time left, then have fun!

Project Description

You will build a website for a financial organization that wants to provide a set of financial calculators for their clients. The home page will provide links to the calculators.

They are interested in having you implement as many of the following calculators you can in the time allotted. Expectations: Getting two done would be good; getting three done would be great.

- A mortgage calculator - it is used to calculate out how much a monthly payment for a loan would be (minus any insurance or taxes), as well as how much interest you would pay over the life of the loan.
    a. It would accept the principal, interest rate, and loan length from the user
    b. It would display the expected monthly payment and total interest paid

    Example: A $53,000 loan at 7.625% interest for 15 years would have a $495.09/mo payment with a total interest of $36,115.99

    This calculator would use a compounded interest formula.

- A calculator that determines the future value of a one-time deposit assuming compound interest - it is used to help you decide how much a CD will be worth when it matures

    a. It would accept the deposit, interest rate, and number of years from the user
    b. It would display the future value and the total interest earned

    Example: If you deposit $1,000 in a CD that earns 1.75% interest and matures in 5 years, your CD's ending balance will be $1,092.62 and you would have earned $92.62 in interest

    Note: The numbers above assume ***daily*** compounding

- A calculator that determines the present value of an ordinary annuity. (Note: hard)
    a. It would accept the monthly payout, expected interest rate, and years to pay out from the user
    b. It would display the present value of that annuity

    Example: To fund an annuity that pays $3,000 ***monthly*** for 20 years and earns an expected 2.5% interest, you would need to invest $566,141.46 today.

NOTE: If your results on any of these calculators are off by a few pennies (not dollars!), don't worry. The difference is likely attributable to rounding and we aren't that concerned about it in this academy.

Hints

Your site won't look like any of the following, but this will give you an idea about what each calculator does:

`https://www.bankrate.com/calculators/managing-debt/annual-percentage-rate-calculator.aspx`

`https://www.nerdwallet.com/article/banking/cd-calculator`

`https://financialmentor.com/calculator/present-value-of-annuity-calculator`

A quick Google search will reveal the formulas needed to make each calculator work or you can read up on some of the formulas at:

Mortgage Payment: `https://www.quora.com/How-is-the-division-of-principal-vs-interest-calculated-on-mortgage-payments`

Future Value: `https://www.gobankingrates.com/banking/cd-rates/how-calculate-cd-accounts-value`
  Note: the `n` in the formula stands for how often the compounding occurs; use 365x per year

Present Value of Annuity: `http://www.1728.org/annuity-presval-formulas.htm`

What Makes a Good Workshop Project?

You should:

- build a consistent look-and-feel throughout the site with intuitive navigation
- implement at least the first two calculators
- have a responsive user interface

You should adhere to best practices such as:

- have a good directory structure (ex: `css`, `images` and `scripts` folders)
- include Bootstrap from a CDN
- have good file naming conventions (ex: lowercase file names with no spaces)
- have well- formatted HTML, CSS and JavaScript (indentions, blank lines, etc.)
- use good names for your HTML elements and JavaScript variables/functions
- use HTML, CSS, and JavaScript comments effectively

Make sure that:

- you use the ESLint tool to ensure you've written good JavaScript!
- you use validators to ensure you don’t have any HTML or CSS errors!
- there are no JavaScript errors at run time (check the Console tab in the browser)

Build a **<u>PUBLIC</u>** GitHub Repo for your code.

- Use an appropriate branch structure and have a commit history with meaningful comments
- Include a README.md file that describes your project and includes screen shots of 1) your home page 2) EACH of the calculator pages you build that shows inputs and correct outputs and 3) one calculator page that shows erroneous inputs and an error message. ALSO make sure to include one interesting piece of code and a description of WHY it is interesting.

Other things we look for as we play with your website include:

- having focus appear automatically on the first input field
- having correct answers with good inputs on each calculator
- displaying monetary values with two digits to the right of the decimal point (and maybe a dollar sign in front of the number if that makes sense)
- the inability to enter bad data because the input is restricted using `<input type="number">`
- having a `reset` button
- using `readonly` on output form fields