# Front-end development

## Module 2

*HTML, Markdown, and Your First Website*

Bits and Bots
study group

# Content

# This module

Welcome to Module 2! Here you will begin to understand the basics of web development. By exploring the essentials of HTML and Markdown, you will soon be equipped with the skills needed to create and publish your very first website. You'll also discover how to showcase your Twine game on your own personal webpage.

In this module the goal is to get familiar with writing content on websites. We will look at two different types of markdown language. The first, HTML, is common across web pages. We will also look at using Markdown on GitHub and programming values and principles (the subject of the expert session this module). Note that the part of programming values and principles, which is at the end of this module, should only be made after you have either attended the expert session or seen the slides from the expert session (we will share these afterwards).

Please remember that Rome wasn't built in a day! It is better to divide the work and exercises that are given in this module. Every part contains a black box with instruction what to do exactly. This might help you to structure the workload.

By the end of this module, your achievements will include:

- Publishing your first website, whether it's a single or multiple pages, utilizing HTML or Markdown.
- Integrating your Twine game into a website, thus showcasing the practical application of your new coding skills.

# Recap of Module 1

Web development refers to the overall process of creating websites or web applications, including the project's design, layout, coding, content creation, and functionality. The front-end is the side of a website that you see and interact with as a user.

**Front-end code** allows users to interact with a website and play videos, expand or minimize images, highlight text and more. The most popular frond-end web development languages are:

- **HTML** (HyperText Markup Language) is the backbone of any webpage. It provides semantic structure and defines elements of a website, such as headings and paragraphs.
- **CSS** (Cascading Style Sheets) is responsible for styling the visual appearance of a website. It allows developers to customize colours, fonts, layouts, and to adapt webpages to different screen sizes.
- **JavaScript** is a dynamic programming language that adds interactive elements to web pages, such as dropdown menus, sliders, forms, and animations.

The **back-end** is the side you do not see when using the internet. It is the digital infrastructure an it looks like a bunch of numbers, letters and symbols that make sure that everything works properly behind-the-scenes. Back-end developers work with systems like servers, operating systems, APIs, and databases and manage code for security,

content and side architecture. Some of the most common back-end web development languages are:

- **Python** is a versatile and more beginner-friendly programming language known for its readability and simplicity. Pythons extensive libraries and frameworks make it a popular choice among back-end developers.
- **Java** is a flexible and widely-used programming language known for its platform independence and scalability. It is commonly used for building enterprise-level web applications that require high performance and security.

***Twine*** is an opensource tool that can be used for telling interactive, nonlinear stories (text-based storytelling). Twine can be used in a web browser or installed onto a computer. Twine's user interface is designed to make it easy to visualize the flow through branches of a narrative (links to different passages in your story). The story can be published to HTML files, which can be uploaded on any web hosting service or shared privately. Games built in Twine can be played without installing extra software. Twine comes with a reference guide and a Cookbook, that contains documentation, example code and explanations.

***GitHub*** is a platform that allows anyone to create, store, manage, and share content. The study group Bits and Bots makes use of GitHub as well. Our repository is used to share learning materials and games that have been created by members. Every repository has a READme file that is written in ***Markdown***. This is an easy-to-read and easy-to-write language for formatting plain text. In GitHub the changes you make are automatically tracked. You can edit, create a branch and merge changes in the main repository. By default a repository has a ***main branch***. By creating your own branch, you can have different versions of a repository at one time. Saved changes are called ***commits***, accompanied by a commit message that explains the changes that were made. The work on different branches will not show up on the main branch, until you ***merge*** it. This can be done with a ***pull request***. We have provided a guide on GitHub, based on the expert session. It can be found here.

# HTML Basics

**What is HTML?**
HTML, or HyperText Markup Language, is the foundational language for crafting web pages. It provides a means for structuring content that browsers can interpret and display.

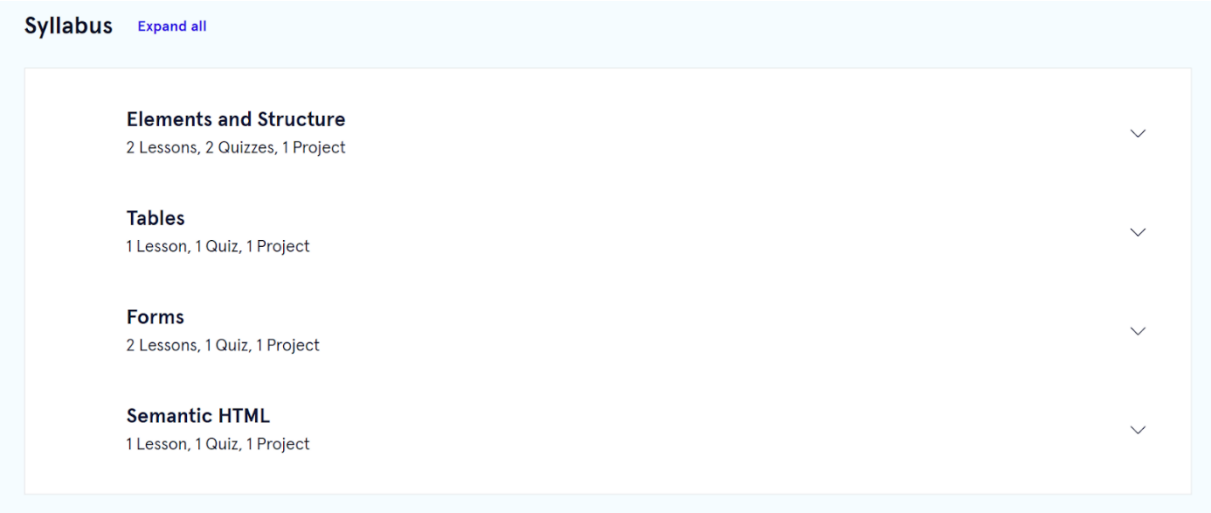
**Key Concepts**
By studying HTML, you will learn how to:

- Define the structure of web pages using elements like headings, paragraphs, and lists.
- Add images and links to create interactive experiences.
- Layer your content for visual appeal and organizational clarity.

HTML can be used by itself to make a webpage but often is used combined with CSS and JavaScript which we will look at later in the course. To understand how the languages

interact, read the following blog: Coding for Web Design 101: How HTML, CSS, and JavaScript Work.

**Learning Resources**
Start by working through the Codecademy 'Learn HTML' course, which is divided into four modules, as you can see in the screenshot below. Everyone should complete at least the introductory module on 'Elements and Structure', but you are very welcome to complete the remaining modules at your discretion, time, and enthusiasm. Note: You do need to log in before you can access this course.




Follow the Codecademy course 'Learn HTML', at least module 1 'Elements and Structure'.

# Creating Your Own Website Using GitHub

## Creating Your First Website

Now that you've learnt HTML you can try writing some yourself using an editor. Remember to start the file with the HTML tag. W3 schools is very useful as a guide to reminding yourself of HTML rules. Once saved as a .html file you should be able to click on the file from your computer and it should open in your browser. It is recommended that you save the main page of the website as index.html. From there you can see how it will appear in the final webpage and make edits. You can also put other files in the same folder such as pictures. Provided everything is embedded correctly it should appear. If you are feeling adventurous you can also create other HTML files for other pages of your website and link them.

Once you are happy with your creation, upload your file(s) to GitHub and create a website, with the following instructions

## Publishing Your Twine Game

You downloaded your Twine game and it consisted of a .html file, as well as maybe some other media. The next steps are simple! Rename the main HTML file of the Twine game downloaded to index.html. Now repeat the steps above and publish it as a website. This will make it easier to share with friends. Do make sure you are comfortable with everything you are publishing to the web however, as these websites will not be private.

Make sure you have fun! This is only a guide and doesn't cover everything. If you want to experiment with different themes or ideas, learn more and play around then please do and share what you find out on our Discord server. There's so many exciting things you can make with a bit of creativity and a good search engine!

# Markdown Basics

## What is Markdown?

Markdown is a lightweight markup language with a plain text formatting syntax. It is a tool used to convert text to HTML for easy content management. You can use it to write ReadMe pages about your repositories in GitHub as well as format GitHub pages. It is simpler and quicker to write than HTML, if you aren't looking to do anything too complicated.

**Learning Resources**

- We will also be following this interactive tutorial here.
- To understand more context of Markdown try this introductory video tutorial on YouTube.
- Use the blog Markdown Crash Course to remind yourself of the basics.

Use the learning resources above, to familiarize yourself with Markdown and try it out with the interactive tutorial.

**Next Steps**

Once you've got the hang of Markdown, create some ReadMe.md files for your repositories to let people know what your repository is about. Learn more about readme files and why they are important here.

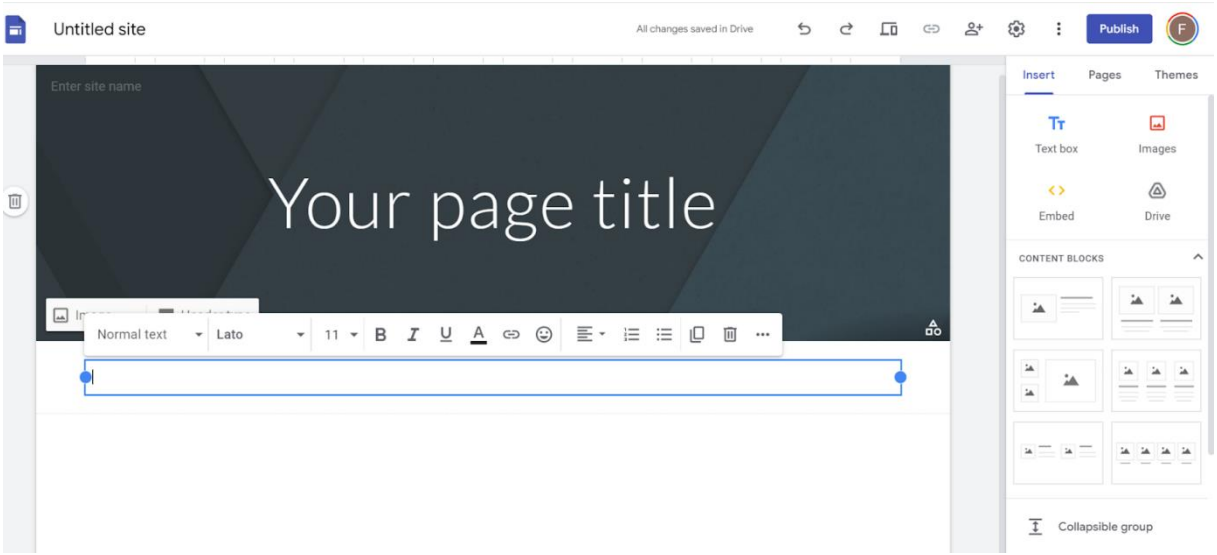
You can try creating a new repository on your GitHub account or use the techniques you've learnt with GitHub pages try making a website using Markdown instead. You will need to upload to your repository either a readme.md file or an index.md file rather than the index.html you created earlier. Remind yourself with the previous guide

Create some READme files in Markdown in your GitHub repositories, including:

- Headers
- Bold and italic text
- A table
- Images

# Creating Your Own Website Using Google Sites

Here is where you can get creative! Google Sites is similar to Google Docs or Google Sheets except you get to publish a website instead. Google Sites provides a user-friendly, drag-and-drop interface for website building, allowing you to create a clean and responsive site without coding knowledge.




You can get as creative as possible without needing any coding languages. However, you may want to use it to host your new Twine game. Clicking the Embed button on the right hand side will allow you to embed your Twine game on the webpage by copying your HTML in. But you could also link between multiple Google pages, embed videos from YouTube (all YouTube videos have an embed button). Here are some instructions on embedding. Here is an example of an escape room style game created with Google pages. Play around or go further and create something to share! Don't forget to keep us posted in Discord.

# Adding Your Website to our GitHub Repository

Now that you've completed the expert session on GitHub, created your first Twine game, understand more about markdown and hosted a page using GitHub you can now share it with everyone!

Following the instructions from the Bits and Bots GitHub expert session add your link in markdown to the readme page here. You can also use the specialised guide on GitHub (written in Markdown).

# Programming values and principles

The expert session in May is about “Programming values and principles” and will be given by Daniel Steinmeier. This part of the module should be done after that expert session. We will also make sure to provide you all the slides of the session afterwards to aid you in the assignment.

**Abstract expert session**
In programming there is a certain notion that some ways of writing code are more elegant than others. We might have an intuitive understanding of the notion of elegance when it comes to natural language. But what is elegance in coding? As we will see this can be the complete opposite of elegance in natural language. Take synonyms for example. Synonyms provide variation and make text less boring. But one of the principles of Python coding is: “There should be one – and preferably only one – obvious way to do it.”

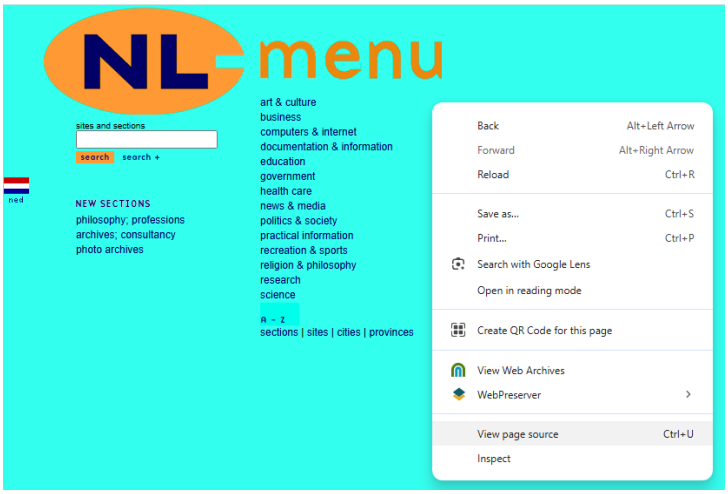
What is this way? In the case of HTML similar principles apply. Why is it 'wrong' to use tables for lay-out? In this session we will look at important coding principles and the values behind them for both Python and the Web. As a sidestep we will also touch upon the question of how generative AI should and should not be used when aspiring to writing quality code.

## Exercise programming values and principles

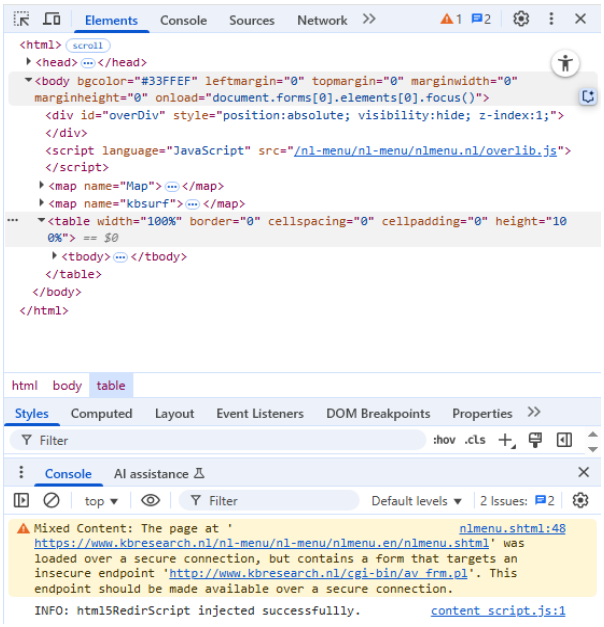
The National Library of the Netherlands (KB) has an old website that was restored from CD-ROM using web archeology techniques. Some parts of the code are about 25 years old. In the meantime a lot has changed, the browser wars are over, web standards have become, let's say, standard practice.

Go and have a look at: https://www.kbresearch.nl/nl-menu/nl-menu/
Then, follow the three steps below.

1. Right-click on the page and view the source-code of the website (you can also use Ctrl + U for this). How many elements can you find that are outdated or not according to current standards? In case you need pointers and also want to experience the struggles of web designers back in the day when web standards were new, please refer to: https://alistapart.com/article/journey/



2. Activate the web-developer toolbar in your browser and go to style editor. For both Chrome and Firefox you can get the style editor by using Ctrl + Shift + i. This will give you the following screen on the right of your browser:




Very few elements are mentioned here because most of it is in the HTML-file itself and not in stylesheets like you would expect of a modern webpage. However, you can still play with the style! Try and change the background color and the color of the text by editing the values in the style editor-window (double-click to edit a specific part). You have to target some elements like body by adding this manually.

3. Add: table, th, td{border:1px solid black;} to the style-editor. Now you see how complex the table lay-out actually is and why this is a difficult method for handling lay-out. Which properties are used for lay-out according to modern web standards?

Make sure to write down your findings. The answers will be provided in the next module.

The developer mode you have been using during this assignment will also be used in future modules to look further into website accessibility, user testing, and standards.