Front-end development

Module 3

HTML and CSS

Bits and Bots
study group

Content

This module

Module 3 will build on some of the fundamentals that you have already learnt such as HTML, GitHub, and the basics of Twine. We will learn about the purpose of CSS and how it can be incorporated into Twine. We will also look at styling GitHub pages and templates. It will also take a quick look at accessibility.

Recap of Module 2

In module 2 you've learnt more about HTML (HyperText Markup Language) basics, with the help of the Codecademy course. It provides a means for structuring content that browsers can interpret and display. HTML can be used by itself to create a webpage but it is often used in combination with CSS and JavaScript (more on those in later modules). Like any programming language, HTML has its own rules and syntax that need to be learned step by step. Every page on the web that you visit is built using a sequence of separate instructions, one after another. Your browser (Firefox, Chrome, Safari, and so on) is a big actor in translating code into something we can see on our screens and even interact with. It can be easy to forget that code without a browser is just a text file. Right-click on any page on the internet, choose "Inspect," and you'll see HTML in a panel of your screen.

HTML uses **tags** to identify different types of content and the purposes they each serve to the webpage. Each type of content on the page is wrapped (surrounded) in tags. For example, for a paragraph, you write the following:

```
<p>This is a paragraph.</p>
```

The key element in HTML to build a website is the **body**. Only content inside the opening and closing body tags can be displayed on the screen. Once the file has a body, many different types of content can be added.

Headings in HTML are similar to headings in other types of media. For example, in newspapers, large headings are typically used to capture a reader's attention. In HTML, there are six different headings, or heading elements. The heading <h1> is used for main headings.

One of the most popular elements in HTML is the ** element**. This is short for division, or a container that divides the page into sections. Divs don't have a visual representation. They can contain any text or other HTML elements, such as links, images, or videos. These sections can be very useful for grouping elements in your HTML together.

To expand an element's tag, you can use an **attribute**. Attributes are content added to the opening tag of an element and can be used in several different ways, from providing information to changing styling. The image tag below, has three attributes: src (source) that specifies the path to the image that is displayed, and two attributes for the size of the image: width and height.

```
<img src="images/minion.png" width="200" height ="450" />
```

Answers Exercise Programming Values and Principles

In module 2, you have made an exercise concerning programming values and principles. In this module we will provide you with the answers of the exercise.

Exercise
The National Library of the Netherlands (KB) has an old website that was restored from CD-ROM using web archaeology 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.

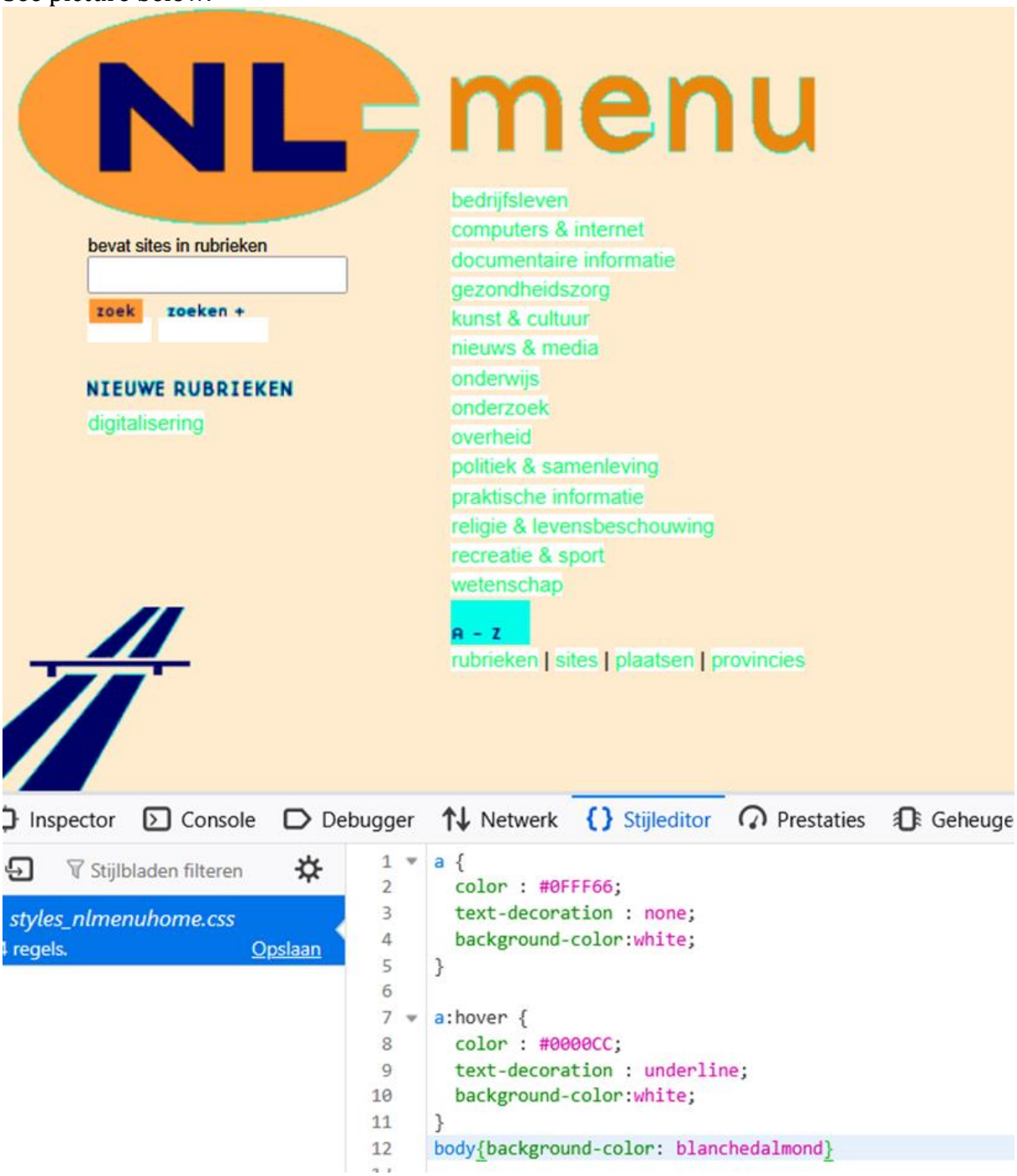
Go and have a look at: https://www.kbresearch.nl/nl-menu/nl-menu/.

Question 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/.

Question 1 - answer
No declaration at the start. JavaScript in the header instead of a separate file. Style like background-colour as inline-attribute on the whole body-element. Inline JavaScript like onload-attribute. Inline CSS like on the div-element. Capital letters in element names. Mouse over behaviour as inline attributes. Empty alt-attributes.

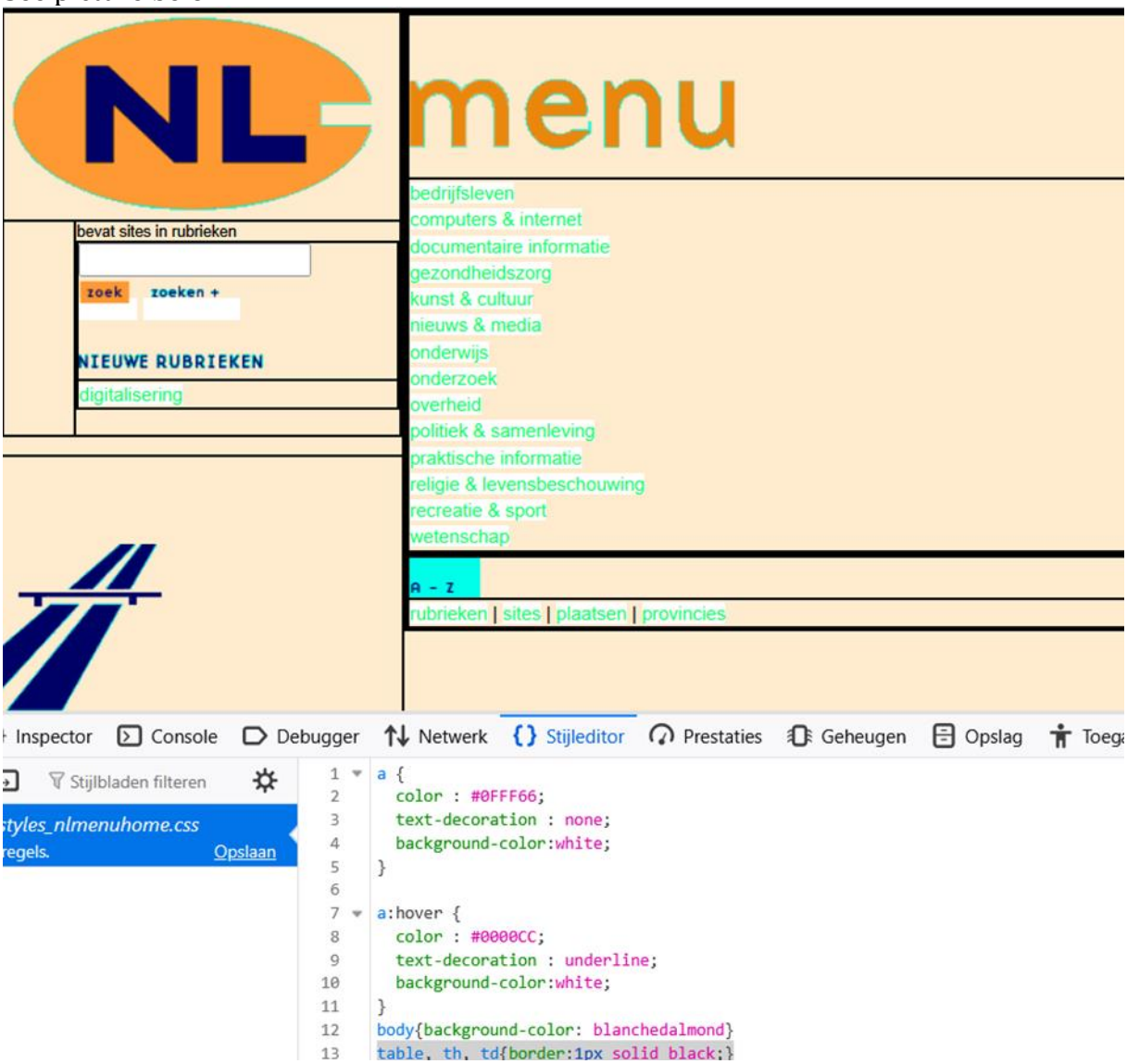
Question 2
Activate the web-developer toolbar in your browser and go to the style editor. For both Chrome and Firefox you can get the style editor by using Ctrl + Shift + i. Very few elements are mentioned here because most of it is in the HTML-file itself and not in stylesheets like you would expect of a modern webpage. However, you can still play with the style! Try and change the background colour and the colour 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.

Question 2 - answer
See picture below.



Question 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?

Question 3 - answer
See picture below.



Properties currently used instead of tables are: float, flexbox and grid, see: https://www.w3schools.com/html/html_layout.asp.

Introduction to CSS

What is CSS?
CSS, which stands for Cascading Style Sheets, is the language we use to style a webpage. CSS is not just about changing colour but is also crucial when making websites accessible (for example readable on mobile and laptop).

Follow the introduction to CSS provided here on the W3 Schools website.

Learning Resources
We recommend starting with the Codecademy 'Learn CSS' course. After completing this course it would be useful to also read this blog and maybe some of the learning resources at the end of it on accessibility in websites. If you still have time and enthusiasm after this, you will notice that there are other CSS courses available, either on different specialist topics or 'Learn Intermediate CSS' courses. The Intermediate course on Codecademy will give you a more advanced overview but if you are interested in a specialist topic and it catches your eye then feel free to go in a different direction.

If you have time, see if you can improve or create something that you have learnt. There are lots of fun resources online that can make your website stand out, so you may want to explore these too. For example the Google Font corpus.

Follow the 'Learn CSS' course on Codecademy.
Note: You do need to log in to access Codecademy courses.

Incorporating CSS with Twine
To start incorporating CSS into your Twine Game we will be following the second and third parts of Adam Hammond's tutorial 'A Total Beginners Guide to Twine 2.1'. These are the sections titled 'Making Your Game Look Awesome with CSS' and 'Adding Videos and Music'.

GitHub and Jekyll

If you have been enjoying using Markdown and GitHub pages then you may be wondering a bit more about how CSS and styling may apply to these. To build on your work around creating pages in GitHub then this section of the GitHub documentation will help. Specifically around creating your own themes and incorporating CSS.

Final Goal for this Module

After working through the resources mentioned in this module, try and incorporate some of these in your game/website. Also try to document what you have used and added somewhere. Do not forget to share your updated games. It is fun to see the games we build grow over time as we learn more.

Extra Challenge: Transitions and Animations Using CSS

CSS not only can be used for styling but also when combined with some HTML it can be used to create animations within your website. This can range from something simple like a shaking button to ants crawling all over the screen! To see some ideas of what can be made take a look. This is all about getting timings, fades and shapes correct. Much of the code can be copied from examples across the web and applied to your webpage but to get an idea of how to create animations from scratch the CodeAcademy course is a good place to start. Two good dictionaries explaining key syntax can be found here and here.

The Command-Line Interface

The expert session in module 3 is focused on the command-line interface. While the expert session will show you the commend-line interface in Windows (the command prompt), we will also make sure to provide you with the commands for the other operating systems (Mac/Linux). In this last part of the module, you're going to apply what you've learned from the expert session yourself. Do not forget to use the command-line interface guide for this exercise. We will share this guide with you after the monthly meeting (option 2).

Exercise Command-Line Interface

For this exercise, you are going to try out some commands and bundle them in a batch script. Follow the steps below. Make sure to only use the command-line interface for steps 2-6.

1. Open the command prompt
2. Navigate your way to a folder
3. Show the content of that folder
4. Make a new folder (in the folder you are at)
5. Duplicate a file and put the duplicate in your newly created folder
6. Calculate a checksum[1] (SHA512 or MD5) of the file in your new folder
7. Create a batch script that includes steps 4-6.
8. Execute that batch script.

You've learnt how to use the interactive shell and save and run your Python programs. It is also possible to run a python file directly from the command prompt. Try it out! Open the command-line interface and execute a Python file.

Extra exercise
Make a list of the most useful commands for you and try to combine them in a batch script.

[1] Checksums are values that are generated from transmitted data before and after transmission. They are a sort of fingerprint for your file. If the checksum stays the same, you know the file is the same. If anything has changed, the checksum too will have changed. This can indicate bit rot or an error that happened during migration.