# Lab 1

In this lab we will learn some basics of HTML and CSS by editing an existing webpage to change its format and presentation. This lab has two parts:

- CSS tasks that change the presentation style of some existing HTML elements.
- HTML tasks that add new HTML elements to the page, along with CSS tasks that format those elements.

## Checking out the project

In order to work with the project, you need a couple of steps steps:

- Start a new repository in GitHub,
- Clone your new repository to a local repository,
- Link your local repository to my GitHub repository that contains startup files,
- Update your GitHub repository with changes from mine.

We will now follow each of these steps

- Start a new repository
  - Register with GitHub[1] if you haven't yet.
  - Once logged in, start a new repository with any name you like (but maybe something like WebAppsLabs would be appropriate). Make sure to set it to Private.
  - Under Settings->Collaborators, add me as a collaborator.
- Clone your new repository to a local repository.
  - Open the GitKraken application, and get it set up if you have not yet. This application helps you move files back and forth between your local computer and the remote GitHub repository that you just forked.
  - In GitKraken, under Preferences->Authentication, use the GitHub.com tab to get GitKraken connected to GitHub.
  - In GitKraken, use the folder icon at the top left and then choose Clone and find your repository under GitHub.com. Make sure the "where to clone" slot has a good path for where you want this project to be created.
  - Choose "Open Now" and "Initialize" when prompted at the top of the application window.
- Link your local repository to *my* Github repository.

[1] github.com

  - On the left side you will see a REMOTES section. Click on the plus sign that shows up when you hover over it. In the resulting window, switch to URL, put "upstream" in the Name space and "https://github.com/skiadas/WebAppsLabsNew.git" in the Pull space. Then click "Add remote".

- Update your GitHub repository with changes from mine.
  - In the main window you should now be seeing a "master" line with a desktop icon next to it (your local repository), as well as a "master" line with my face on it (my remote repository). It should contain two "commits" to it, the last one of those being "Initial commit". Right-click on it and choose "Reset master to this commit (Hard)".
  - You should now be seeing only two commit lines, with the second one being the desktop master. Right-click the top one (my master) and choose "fast-forward master to upstream/master". You should now see the two masters line up.
  - Click on Push near the top, and click Submit in the pop-up message. You should now see a third icon, corresponding to your remote, pop up.
  - Back on the GitHub webpage, reload the page and you should now see some folders and files present.

Now you have a local copy of your project. To work with it, you should do two things:

- Start the SublimeText application and use the Open menu to select the folder that was created by the clone process above. You should see a Lab1 folder in it, which contains the files you need to change.
- Find the sample.html file in this Lab1 folder and open it up on your web browser.

You will be editing this project for your lab assignments. When you are ready to submit your assignment, follow the following steps:

- Make sure the files are saved in SublimeText.
- In GitKraken, open the project. You should see on the right side the two "unstaged files" that you edited. You may click on them to visually see all the changes you did to those files.
- Click the "Stage all changes" button to add all those files to your next "commit". You should see them move to the "staged files" section.
- At the "Commit Message" section at the bottom, type a simple Summary, like "Lab 1 Submission". Then click the big Commit button at the bottom. You should see a new step created in the "tree" in the middle panel. You have just created a local "commit" of your changes. This protects them from being lost or accidentally deleted.
- Click the "Push" button at the top of the file, to move these changes to your remote repository in GitHub.com. You can check that this has happened if you look at that repository on your web browser.

To test this process, DO this right away! Add your name at the appropriate spots in the two files (html and css), save and create and push a commit as described above.

## Details on the CSS tasks

For this part of the lab, you should have sample.html open in the browser, and you should be editing ourModifications.css. We have provided all the "selectors" you will need, you will just need to fill in the rules as per the instructions. You will need to consult the CSS reference[2] along the way.

A lot of these ask you to set colors. You can use any number[3] of color pickers[4] that are available online.

Check your progress by refreshing the html page on the web-browser and seeing how it looks.

1. There is a rule that catches the element with id of main. In that rule you will need to:
   - set the margins to be 20% on left/right and 10% on top/bottom.
   - set a border of 1px thickness, solid, and with a color in the gray area, perhaps "#DDD" (look in the syntax area on this link[5] for different color specification formats).
   - add a padding of 2 ems on all sides.
2. There is a level 3 heading rule. In it you will need to:
   - set it to align the text to "center".
   - set its font size to 200%.
   - put a border at its bottom, dotted, 2px, and of some gray-ish color.
3. There is a rule for list items that are inside of the element with id main. In that rule:
   - set the font size to 115%.
   - add padding of 0.2em all around.
   - instruct it to not use the bullet on the left (list-style-type is the relevant argument).
   - Set its width to 100%.
   - Add a 3px margin at top and bottom, and 0 left-right
   - Add a 2px solid black border.
   - Add a 10px border radius. You should get a nice rounded corners effect.
4. There is a rule that catches the items with a class of "remove" that are within a list item. In that rule:
   - set the element to float to the right.
   - set its line height property to 0.9 ems. This should make it a bit more centered vertically.

[2] https://developer.mozilla.org/en-US/docs/Web/CSS/Reference
[3] http://paletton.com/
[4] http://www.colorpicker.com/
[5] https://developer.mozilla.org/en-US/docs/Web/CSS/color

5. There is a rule that matches the list items when one hovers over them with the mouse. In that rule:
    - set the background color to be a light ciel, maybe something like “#AEE”.
6. There is a rule targeting the “ul” element that is the list. In that rule:
    - set the padding equal to 0 to reset one of the browser defaults.
7. There is a rule targeting an element with class “edit”. In that rule:
    - set the width of the element to 100%.
8. There are two rules targetting buttons, one targetting them all the time, and one targetting them *only* when you hover.
    - In the rule that holds always, set a 2px solid black border, and a background color of your choosing. Also give them a 10px border radius.
    - In the rule that holds on hover, choose a different background color, and also change the color of the border.

## Details on the HTML tasks

In this part of the assignment we will practice by adding new HTML elements into the sample.html file. Namely, our goal is to create a table of the tasks.

1. Start a new div tag below the previous main div tag, and set its id to "taskTable".
2. We will now create a table into this tag. A table starts with a table tag, so create such a tag inside your newly created div tag.
3. The first tag inside a table is typically the tag that specifies the headings of the table. This is called a thead tag. Go ahead and add this tag inside the table tag now.
4. Inside the thead tag create a tr tag. This creates a “table row”.
5. Inside this tr tag, add three th tags one after the other, whose contents say “Task”, “Due Date” and “Completed”. You should now see the three headings show up at the bottom left of the screen, this is where our table shows up right now.
6. After the closing of the thead tag but within the table tag, start a tbody tag. This will contain the data rows of our table.
7. In this tbody tag, add at least four tr tags, corresponding to four tasks. Each tr tag should contain three td tags, the first tag containing the task’s text, the second containing a due date and the third containing “yes” or “no” depending on whether the tasks is completed or not. Choose whatever 4 tasks you want. You should now be able to see these rows of data in your table.
8. To the opening tr tags for your four data rows, add class="..." attributes to them as follows: If the due date of the task is within the next 3 days, add the class “urgent”. If the task is completed, add the class “completed”. (you may need to add both in some cases, separated by space). If none of your task due dates and completions fit into this, change them to make things a bit more interesting. It is OK if some of your tasks don’t get any class attribute.

Now we will add some css rules to format this table a bit. The following should be added near the bottom of the css file.

1. Create a new rule that targets the table element. In it:
   - Set the left-right margins to 20%, and the top/bottom margins to 10%;
   - Add a 1px solid gray background to it.
2. Create a rule that targets all td and all th elements (you can do this by separating them with comma). In it:
   - Add a 0.5em padding all around.
   - Add a 1px solid black border.
3. Create a rule that targets all th elements. In it:
   - Add a very light gray background color.
4. Notice the very light white spacing around the cells. This is a special behavior for tables. To disable it, go back to the table rule you created a couple of steps above and add a border−collapse: collapse; entry in it. You should see that slight spacing disappear.
5. We now want to color the entries based on whether they are urgent or not. Create a "td.urgent" rule, which targets the urgent row entries.
   - Set the background color of those cells to some sort of red.
   - Set the font-weight to bold.
6. Set a rule that targets the tr rows that are completed.
   - Set the font-style to italic.
   - Set the color to some light grey.
7. Add a hover effect on the td elements, to change their background color when we hover over them. You will notice this works even for the urgent rows. Think about why that is. How does the browser decide which of the two background colors to use.