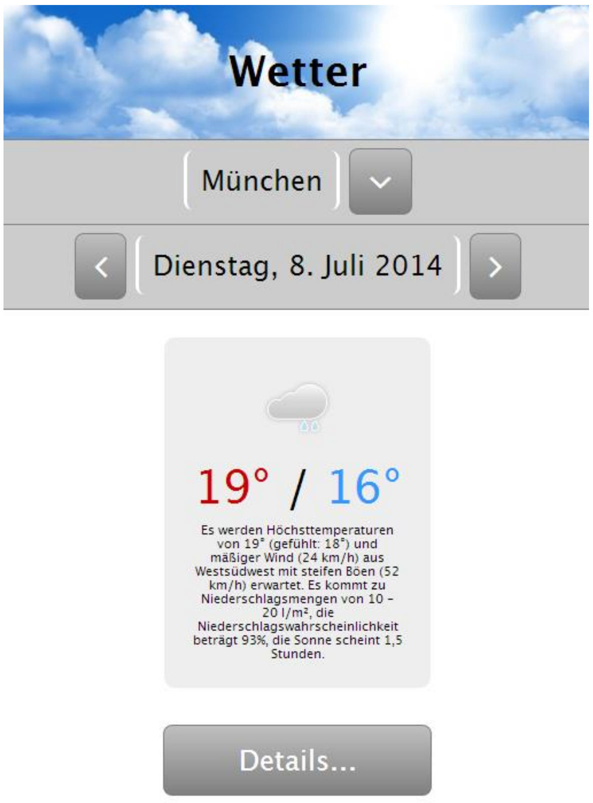
Final Exam

During this exam, you'll create a weather app prototype.

> **Begin**: 14:00h. Download the file "exam.zip" from Moodle, which contains a document structure, images and JS frameworks: https://www.moodle.tum.de/mod/resource/view.php?id=220858
>
> **End**: 16:00h. Upload your solution to Moodle at the end of the session: https://www.moodle.tum.de/mod/assign/view.php?id=220863
>
> **Total Points**: 95 P.

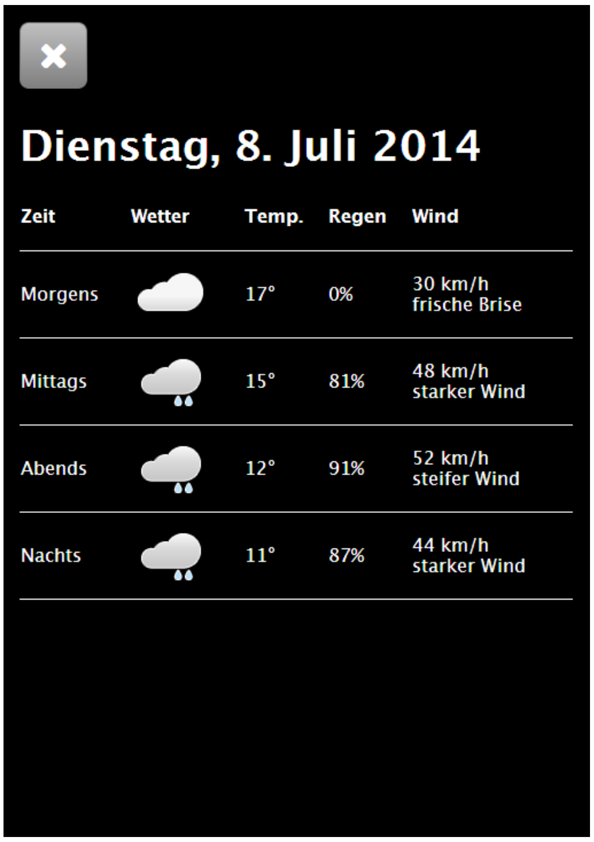
Please, make sure that the layout of your prototype corresponds to this sample below:

View 1: Weather overview



Es werden Höchsttemperaturen von 19° (gefühlt: 18°) und mäßiger Wind (24 km/h) aus Westsüdwest mit steifen Böen (52 km/h) erwartet. Es kommt zu Niederschlagsmengen von 10 - 20 l/m², die Niederschlagswahrscheinlichkeit beträgt 93%, die Sonne scheint 1,5 Stunden.

View 2: Weather details



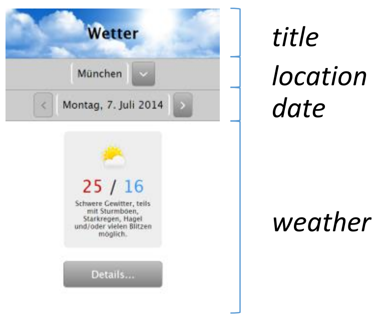
Dienstag, 8. Juli 2014

Zeit	Wetter	Temp.	Regen	Wind
Morgens		17°	0%	30 km/h frische Brise
Mittags		15°	81%	48 km/h starker Wind
Abends		12°	91%	52 km/h steifer Wind
Nachts		11°	87%	44 km/h starker Wind

Please fulfil the following requirements.

HTML Structure (20 Points)

Requirement 1: Start with the provided “index.html” template. Create the four structural elements *title*, *location*, *date* and *weather* with their correspondent ids. (2 P)


Wetter
München
Montag, 7. Juli 2014
25 / 16
Schwere Gewitter, teils mit Sturmböen, Starkregen, Hagel und/oder vielen Blitzen möglich.
Details...
title
location
date
weather

Requirement 2: Please, insert the “Wetter” headline at the topmost section. (1 P)

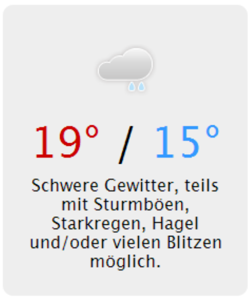
Requirement 3: Create placeholders for the location selector München and the date selector Dienstag, 8. Juli 2014 in the corresponding sections. Those need to be inline elements of the class *select* and have an id. The white brackets will be created via CSS (no text). (2 P)

Requirement 4: Create empty inline elements for the buttons ⌄, ‹ and ›. Assign the CSS class *button*. For adding the arrows, proceed in the following way: (1 P)

Add the following CSS classes (present in the included Font-Awesome library, http://fortawesome.github.io/Font-Awesome/) besides *button*. The corresponding icons should appear (in Firefox and Chrome):

- *fa fa-angle-down fa-lg*
- *fa fa-angle-left fa-lg*
- *fa fa-angle-right fa-lg*

Requirement 5: Create a block level element with the id *info* inside the weather section: (1 P)


19° / 15°
Schwere Gewitter, teils mit Sturmböen, Starkregen, Hagel und/oder vielen Blitzen möglich.

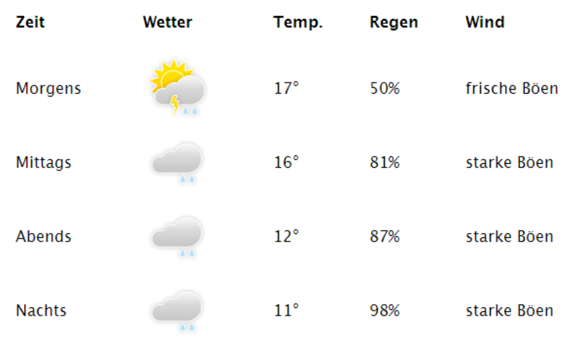
Requirement 6: Create the elements for *image*, *temperature*, and *text* inside the info box. Use dummy text and the images in the folders “weather”. Each of the elements need to be addressable via CSS/JS. Minimum and maximum temperature need to be styled separately later on. (4 P)

Requirement 7: Create a block level element inside the weather section (but outside the info panel) for the Details... panel. It needs to be of the class *button* and contain “Details…” as text. (1 P)

Requirement 8: Your app has to consist of only one document (containing view 1 and view 2). Add a *overlay* section (with corresponding id) in your body after all previous sections (1 P)

Requirement 9: Add another inline element (for the close button, Font-Awesome symbol is “fa-times”), a headline (for the date) and a table (id: *details*) to the *overlay*. (3 P)

Requirement 10: Define the table cells. Use <th> in the first row. Fill the table with dummy content. The table CSS is already given: (4 P)



Zeit	Wetter	Temp.	Regen	Wind
Morgens	⛈	17°	50%	frische Böen
Mittags	🌧	16°	81%	starke Böen
Abends	🌧	12°	87%	starke Böen
Nachts	🌧	11°	98%	starke Böen

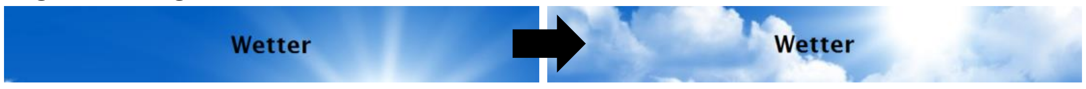
CSS Layout & Style (35 Points)

Requirement 11: For the overview screen, modify the *title*, *location* and *date* sections in CSS: (4 P)

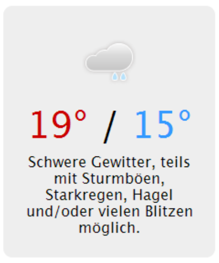
- Increase the distance between content and border
- Contents (text, buttons and selectors) have to appear in the centre of the screen
- The background is light grey
- Each of the sections title, location and date is followed by a dark grey bottom line

Requirement 12: Cloudy title background. (3 P)

- Overwrite (only) the *title* section to display the image “background.jpg” provided in the folder “images”.
- The background always has to keep the size of the browser window. Keep the image’s aspect ratio in the height.
- Align the background to the centre to show both sun and clouds:



Requirement 13: Style the info panel with the following requirements: (7 P)



19° / 15°

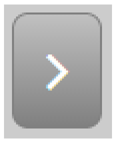
Schwere Gewitter, teils mit Sturmböen, Starkregen, Hagel und/oder vielen Blitzen möglich.

- Width: 200 Pixel
- Distance between outer top and bottom elements: 24 Pixel
- Centred (tipp: left and right outer distance)
- Light (not white) background colour
- Rounded corners
- Distance to elements inside: 16 Pixel
- Text inside: centred

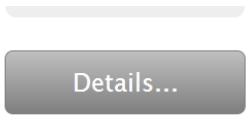
Requirement 14: Style the elements inside the info panel with the following requirements: (4 P)

- The distances around all elements are increased to 8 Pixel
- The descriptive text "Schwere Gewitter…" has a smaller font size
- The temperature "19° / 15°" has a larger font size
- The maximum and minimum temperature is styled red respectively blue.

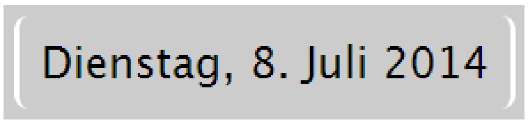
Requirement 15: Style the buttons to fulfil the following criteria: grey border, round corners, gradient background (silver to grey), increased distance between text and border, white text/icon colour, pointer as cursor, centred text. (7 P)



Requirement 16: Layout (only) the button in the weather section to have a width of 200 Pixel and appear centred. (2 P)



Requirement 17: Style the selects' border: (2 P)



Requirement 18: Bring the *overlay* to a position over the other app contents:

- The *overlay* fills the whole screen (2 P)
- It has a black background (1 P)
- It has white text (1 P)
- Increase the padding to 16 Pixel. Ensure that the overlay size is still maximized (no scroll bars) (1 P)

Requirement 19: Modify the *overlay's* CSS so it's no longer displayed. (1 P)

JS/jQuery Interaction (40 Points)

For the interaction part, you may generally use JS and/or jQuery.

Requirement 20: The overlay can be opened and closed

- Add an event handler to the "Details…" button to show the overlay section when clicking.(2 P)
- Add an event handler to the "X" button to hide the overlay section when clicking. (2 P)

Requirement 21: Changing the days is possible

- Create a global variable `currentDay` and initialize it with 0 (1 P)
- Create two functions `previousDay()` and `nextDay()` in "javascript.js" (1 P)
- Call them in the click events of the left and right buttons. (1 P)
- Increase/decrease the value of `currentDay` in the two functions (1 P)

- Have a look at the file "weather.js". There you find a definition of the global array `dates`.
- In the onclick handlers, modify the *date* selector according to the currently selected day (`currentDay`; retrieve the text from the array `dates`). (2 P)
- Ensure (using if conditions) that the minimum and maximum day (array elements and positions) are not exceeded. Use the actual array size (not the constant 7). (2 P)

Requirement 22: Modify the event handlers to work with the global array `weather` (defined in "weather.js" instead of the array `dates`. Note: this now contains objects. (3 P)

Requirement 23: Create a function `loadInformation()`. Its purpose is to load the weather information of the selected day into the info panel. Implement the functionality and call it at a suitable place in your previous/next event handlers. Tipp: Don't forget to add the file path and extension ".png" to the image name. (5 P)

Requirement 24: Load the details.

- Build the *details* table's contents (header rows and data rows) programmatically (Note: the *detail* property of the objects in the `weather` array contains 4 objects in an array with the necessary data). (5 P)
- Use a for loop to assemble the details information for each daytime (2 P)
- Do it in a function `buildDetailsTable()`, return a String of the contents. (2 P)
- Call the function within `loadInformation()` and replace the *details* table's contents with its result. (1 P)
- Update the date field in the overlay (1 P)

Requirement 25: Write a function `convertDaytime(hour)` which converts an arbitrary hour between 0 and 24 to the four daytimes "Morgens", "Mittags", "Abends", "Nachts" and apply it in the details table. (4 P)

Requirement 26: Convert the windspeed (which is given in km/h in the objects) to readable text in the details table.

- Create a function which accepts v as an argument and returns a string representation of the wind strength (2 P)
- Retrieve the names of the wind strengths from the global array `beaufort` on a scale between 0 and 12 (see weather.js) (1 P)
- A Windspeed in km/h is converted by $Beaufort = \left(\frac{v}{3{,}010\ km/h}\right)^{2/3}$. (2 P)