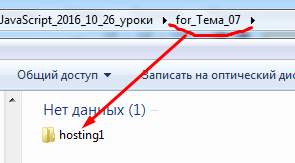
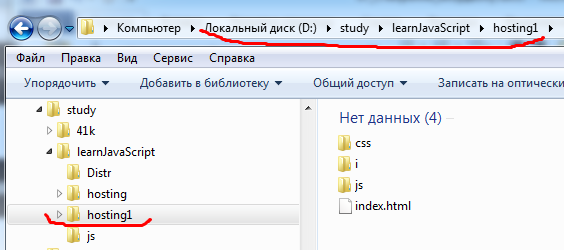
**Тема 7. Створення акордеону**

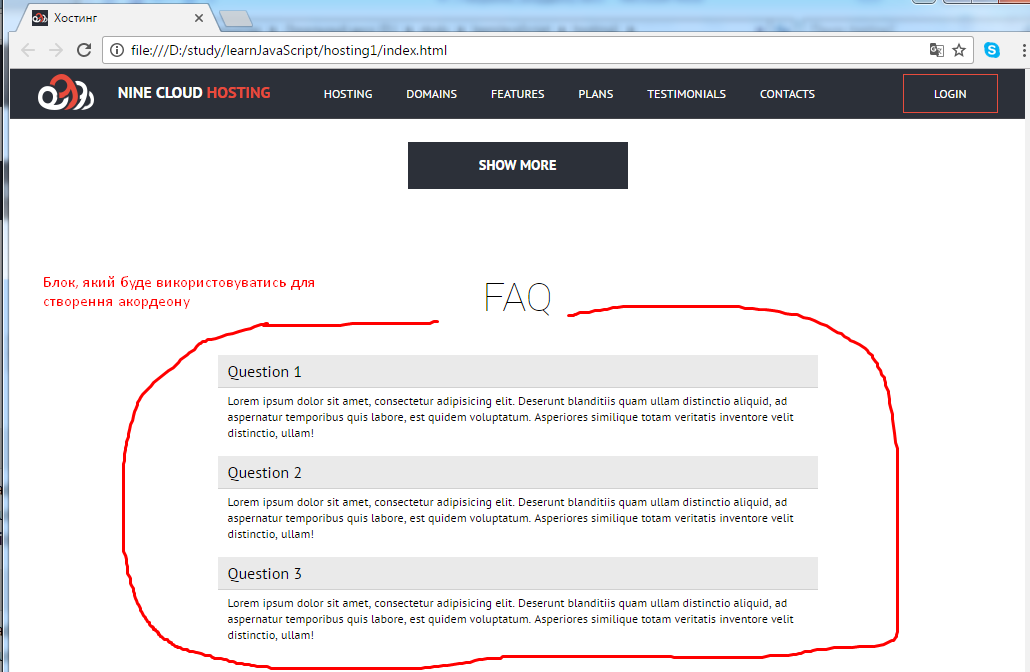
На цьому занятті ми вивчимо декілька нових методів і навчимось створювати акордеон.

Для роботи скопіюємо папку

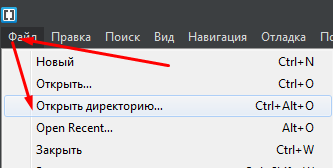


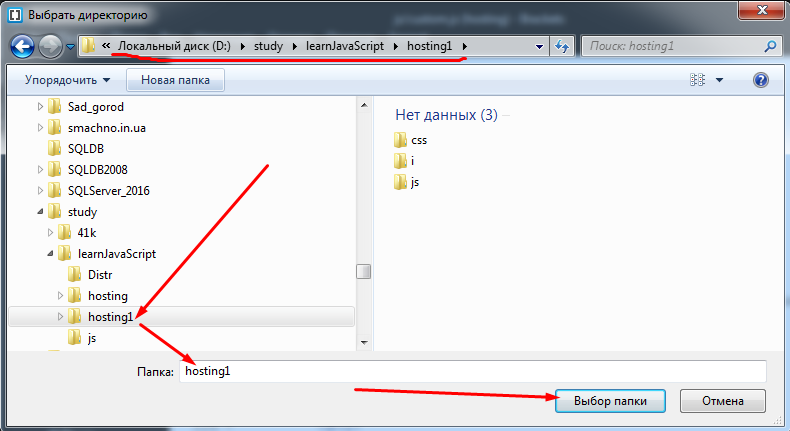


В цій папці доданий блок, який буде нами використовуватись для створення акордеону. Внизу можемо бачити

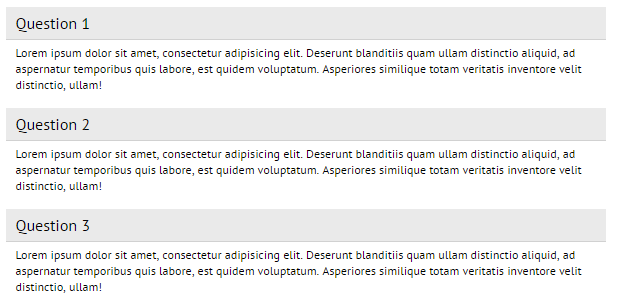


Змінюємо робочу папку в редакторі brackets

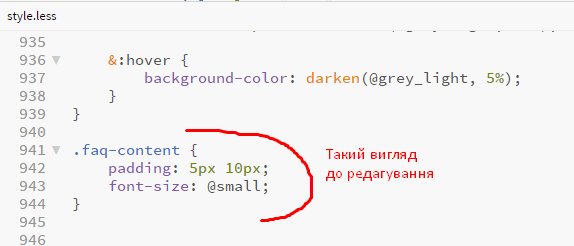




Ідея полягає в тому, що ми будемо бачити лише запитання. При кліку по запитаннях буде відкриватися контент.



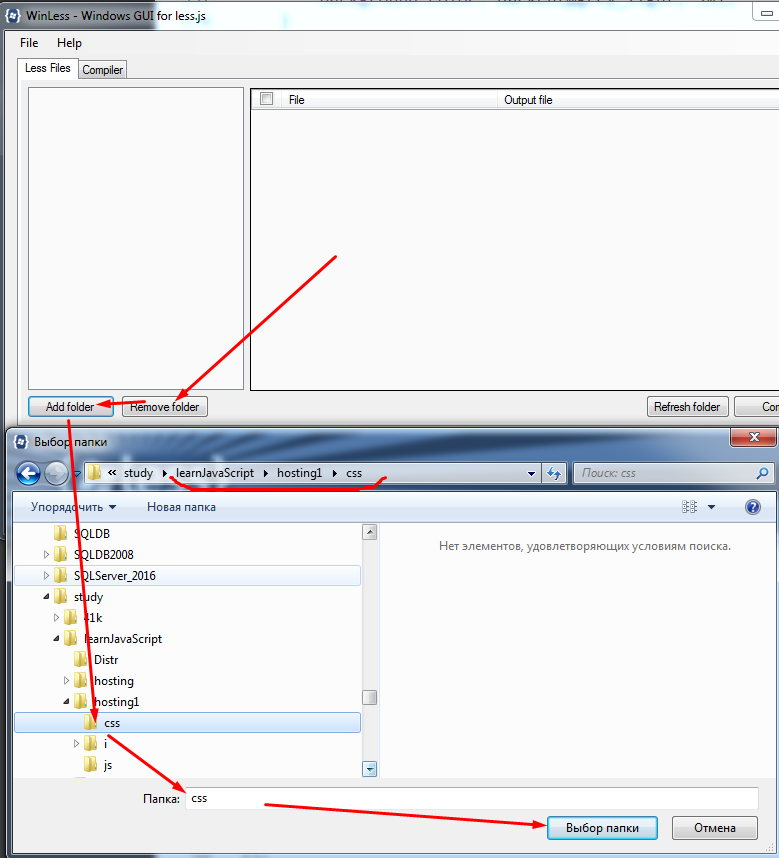
У файлі style.less відключимо показ контенту.

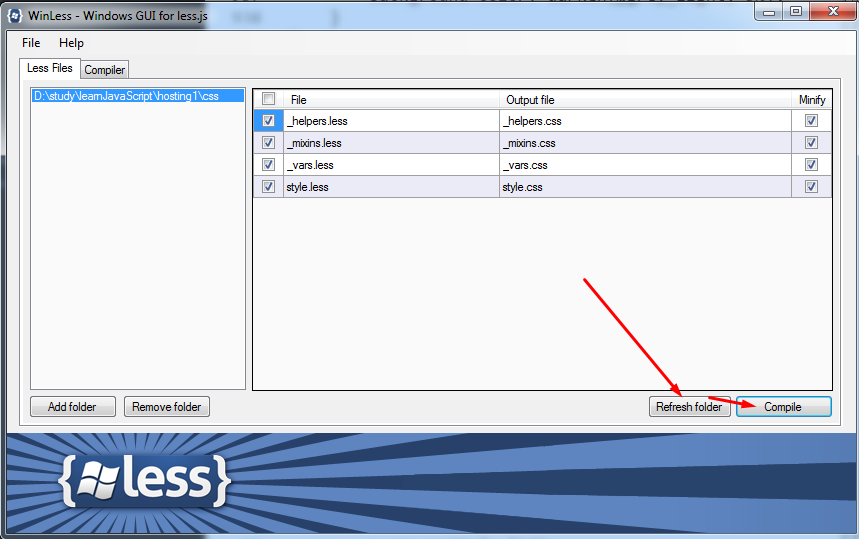


Дописуємо рядок 944, який відключає показ.

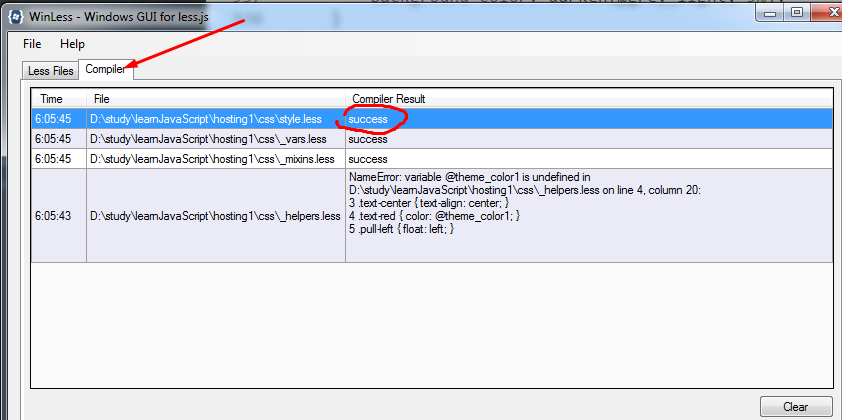


Виконуємо компіляцію.

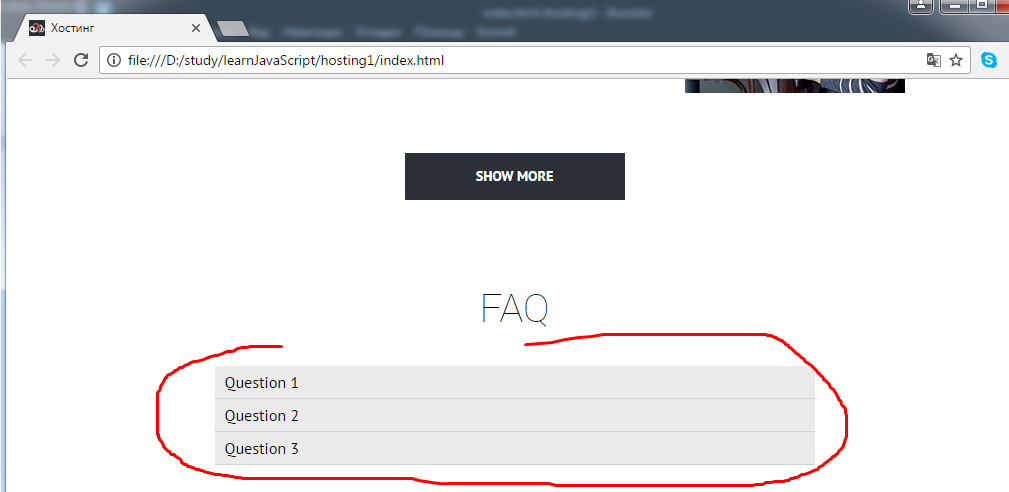




Можемо переконатись, що style.less скомпілювався успішно.



Обновляємо контент браузера. Маємо



Отже, відображаються лише заголовки.

Далі переходимо до файлу index.html.



Тут ми бачимо 3 запитання. В кожному запитанні є title і content.

Контент ми приховали. Відображається лише title.

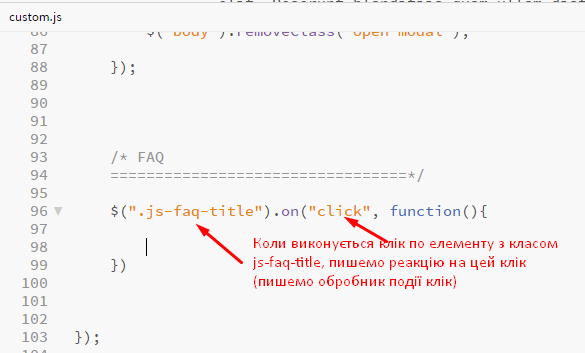
Тайтлу ми повинні присвоїти якийсь js-клас, для того щоб можна було відловлювати кліки (подію click).

Отже, ми для кожного заголовка запитання присвоюємо клас



Далі треба відловити клік по цьому класу.

Щоб це реалізувати, ми у файлі custom.js створюємо окремий розділ.



Оскільки заголовок насправді є посиланням, то в першу чергу відмінимо стандартну поведінку посилань.



Далі треба реалізувати механізм, який буде показувати контент.

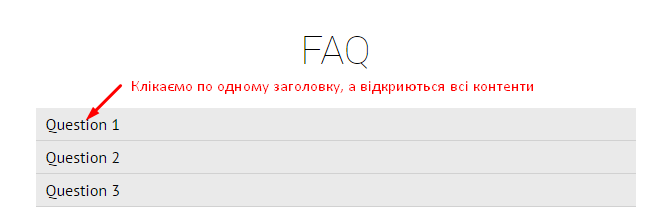
Спочатку кожному контенту присвоїмо клас.

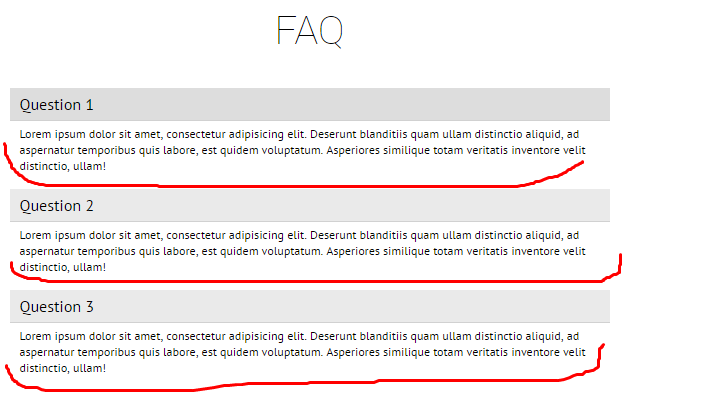


Далі зробимо вибірку цього контенту (по імені класу). І зробимо просто fadeIn() – показати. Подивимось, що вийде.



Переглянемо в браузері.





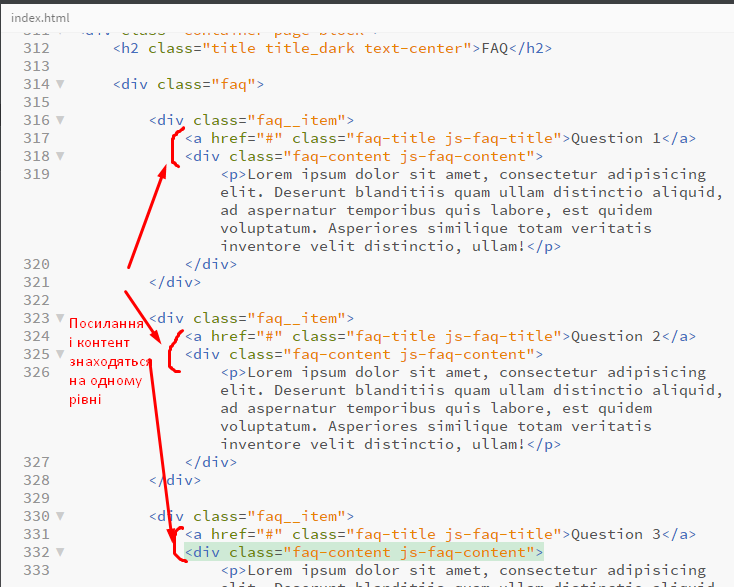
Все вірно. В рядку 100



ми написали, що треба знайти всі елементи з класом js-faq-content, а потім їх показати. Як написали код, так він і виконався: у нас 3 елементи з класом js-faq-content, всі вони і відобразилися.

**Завдання** – зробити так, щоб відкривався саме той контент, що відноситься до заголовка, який ми вибрали.

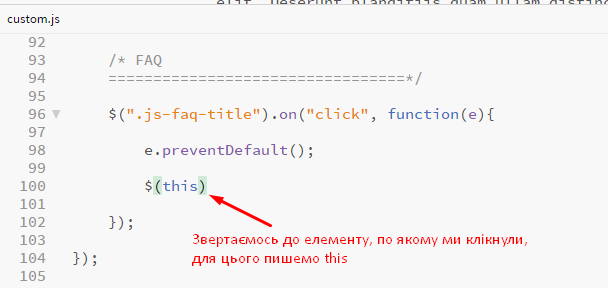
Можна звернути увагу на те, що посилання і контент (які пов’язані між собою) – знаходяться на одному рівні в DOM-дереві. Тобто контент іде відразу після посилання (і все це знаходиться на одному рівні).



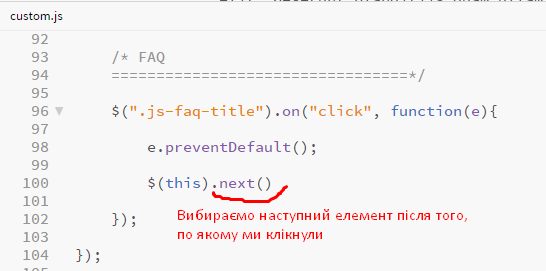
А значить ми можемо використати метод **next()**

Метод next() дозволяє одержати наступний елемент після поточного.

Отже, ми звертаємось до елементу, на який ми клікнули, пишемо **this**



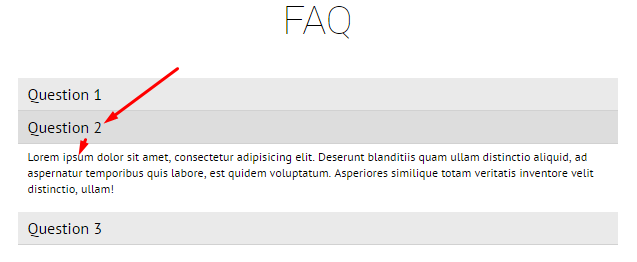
А далі пишемо .next() – для того, щоб звернутись до елемента, який іде після поточного.



Ну а далі допишемо метод, щоб показати даний елемент.

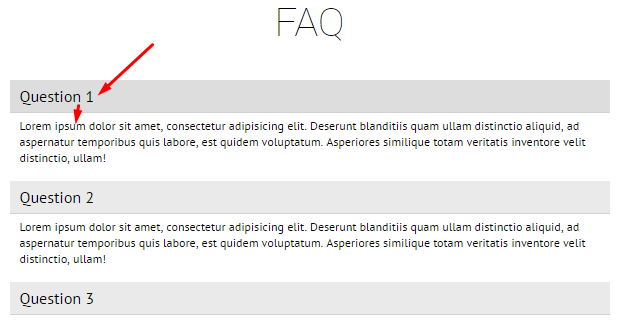


Перевіряємо в браузері.



По якому заголовку клікнули, той контент і відобразився.

Клікнемо, наприклад, ще по заголовку першому.

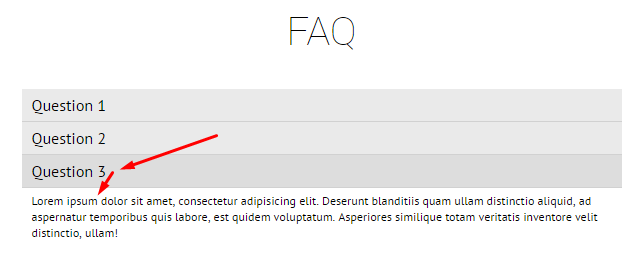


Бачимо, що відобразився контент і для першого заголовка.

Але fadeIn() не зовсім підходить для акордеону.

Треба, щоб контент плавно виїжджав. Для цього використовується метод **slideDown()**





Дійсно, через slideDown() контент плавно виїжджає.

Цей метод також має параметри fast і slow.



Перевіряємо значення fast і slow. Також можна задавати значення в мілісекундах (з якою швидкістю відкриватися).



Коли ми вказали значення 2000, то відкриття відбувається надто повільно.

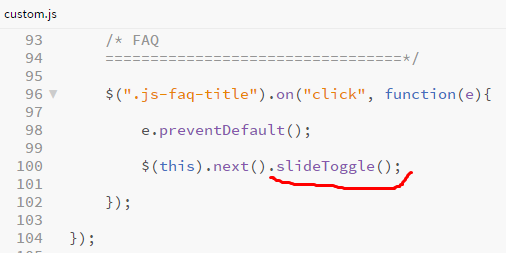
Замінимо значення на slow.

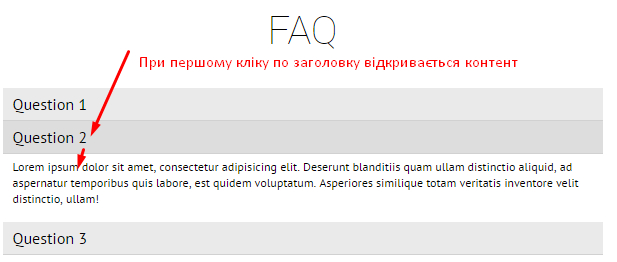


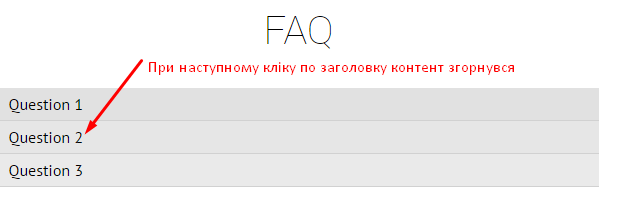
З відкриттям контенту ми розібрались.

Треба зробити так, щоб контент при повторному кліку по заголовку міг закриватись. Для закриття існує метод **slideUp()**

Але краще використовувати **slideToggle()** – він поєднує в собі slideDown() і slideUp(). При цьому перший клік відкриє контент, наступний клік – закриє.







Якби навпаки, в початковому стані елементи були відкриті, то перший клік закрив би контент, а наступний клік – контент відкрив би.

**Для даного прикладу акордеон ми зробили**

Дещо ускладнимо завдання. Нехай ми відкрили 1-ий контент. Далі, клікаючи по заголовку 2, ми хочемо, щоб контент 1 закрився, а контент 2 відкрився.

Для цього, якщо ми клікаємо по посиланню, нам треба додати для нього якийсь клас.

Але спочатку створимо змінну, яка вказує по якому елементу ми клікнули (це прискорює роботу).



Щоб додати клас, ми знаємо метод .addClass().

Щоб вилучити клас, існує метод .removeClass().

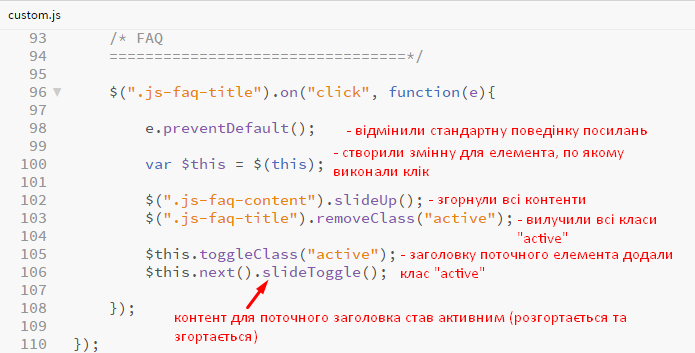
А ще є поєднання цих методів – це метод **.toggleClass().** А далі в параметрах вказуємо, який клас ми хочемо додавати чи вилучати.

Нам треба зробити так, що при кліку по заголовку контенти всіх інших заголовків приховувалися, а контент, що відноситься до поточного заголовку навпаки виводився.

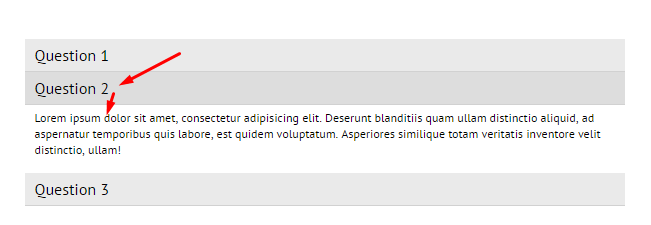


А далі у всіх заголовків вилучити клас «active».

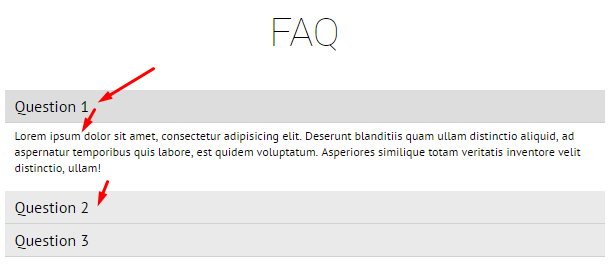




Перевіримо в браузері.



Коли клікаємо по першому елементу



то контент першого заголовка відобразився і автоматично згорнувся контент інших елементів (в даному випадку 2-го елемента).

Але є проблема. Якщо ми клікнемо двічі підряд по тому самому заголовку, то робота виконується не зовсім коректно.

Нам треба робити перевірку («якщо у даного елементу немає класу “active” …)



Перевіряємо в браузері – все працює.

Ще раз переглянемо код.

рядок 96 – при кліку на title

рядок 98 – відміняємо стандартну поведінку посилань

рядок 100 – створюємо змінну для елемента по якому виконали клік (це прискорює роботу модуля).

рядок 102 – робиться перевірки, чи немає в поточного елемента класу “active”. (знак оклику означає заперечення, has – має, ! і has – не має).

рядок 103 – згортаємо всі блок з контентом

рядок 104 – вилучаємо у всіх заголовків клас “active”

рядок 107 – додаємо клас “active” саме для того заголовка, по якому ми клікнули

рядок 108 – відкриваємо/закриваємо контент для поточного заголовка