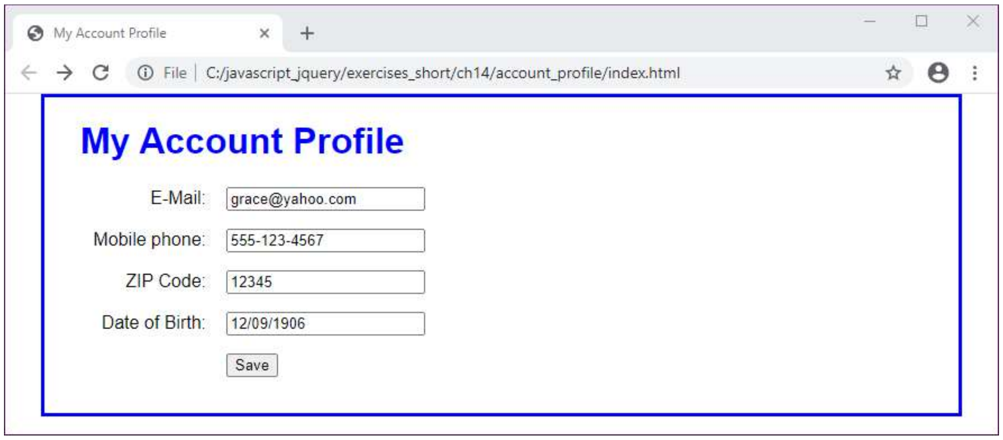
Short 14-1 View the query string of a URL

In this exercise, you'll adjust the Account Profile application so it displays the data that is in the query string of the URL. Estimated time: 5 to 10 minutes.

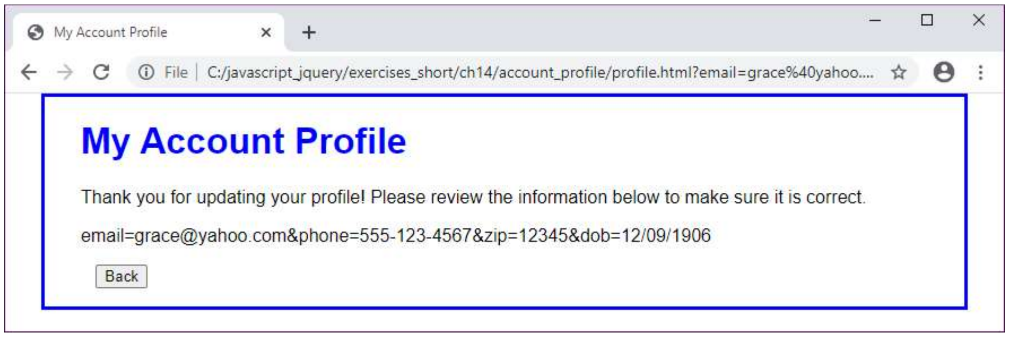
Here is the enhanced application, with account profile information about to be saved:


My Account Profile
File
C:/javascript_jquery/exercises_short/ch14/account_profile/index.html
Save

My Account Profile

E-Mail:	grace@yahoo.com
Mobile phone:	555-123-4567
ZIP Code:	12345
Date of Birth:	12/09/1906

And here is the second page of the application after the profile is saved. Notice that the profile data is in the URL, and special characters like "@" and "/" are encoded:


My Account Profile
File
C:/javascript_jquery/exercises_short/ch14/account_profile/profile.html?email=grace%40yahoo....
Back

My Account Profile

Thank you for updating your profile! Please review the information below to make sure it is correct.

email=grace@yahoo.com&phone=555-123-4567&zip=12345&dob=12/09/1906

1. Open the application in this folder:

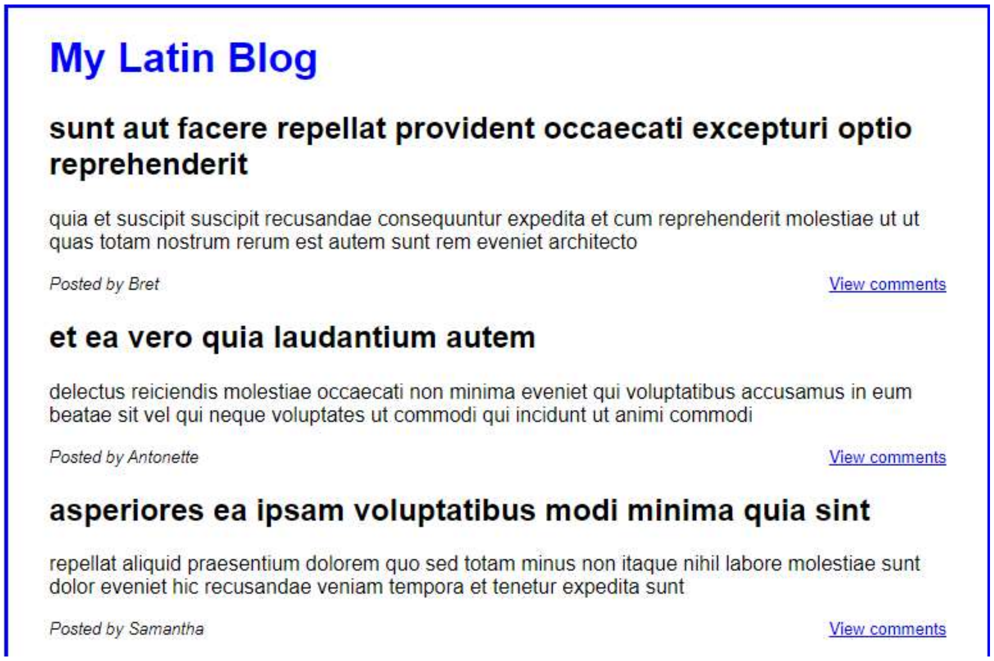
   `exercises_short\ch14\account_profile`

2. Review the index.html file, and notice that the elements on the page are coded within a form element with its method attribute set to "get" and its action attribute set to "profile.html". Then, review the JavaScript file, and notice that when the validation checks pass, the form is submitted. This is how the form data gets in the query string.

3. In the embedded JavaScript for the profile.html file, use the location object to retrieve the query string value. Remove the question mark from that value and then display the value in the span element whose id is "profile". Use the decodeURIComponent() function so the special characters display correctly.

Short 18-1 Enhance the Ajax for an application

In this exercise, you'll update a Blog application to change the way the blog posts are displayed. When you're done, the application should look and work the same as it did before, but it should display different blog posts. Estimated time: 15 to 30 minutes.



My Latin Blog

sunt aut facere repellat provident occaecati excepturi optio reprehenderit

quia et suscipit suscipit recusandae consequuntur expedita et cum reprehenderit molestiae ut ut quas totam nostrum rerum est autem sunt rem eveniet architecto

Posted by Bret View comments

et ea vero quia laudantium autem

delectus reiciendis molestiae occaecati non minima eveniet qui voluptatibus accusamus in eum beatae sit vel qui neque voluptates ut commodi qui incidunt ut animi commodi

Posted by Antonette View comments

asperiores ea ipsam voluptatibus modi minima quia sint

repellat aliquid praesentium dolorem quo sed totam minus non itaque nihil labore molestiae sunt dolor eveniet hic recusandae veniam tempora et tenetur expedita sunt

Posted by Samantha View comments

1. Open the application in this folder:

   `exercises_short\ch18\blog\`

   Run and test the application.

   Note that it displays five blog posts. Note that each post includes the name of the user (in this case, all posts were made by Bret). And note that a "View comments" link allows you to view the comments for the post.

2. Review the JavaScript file. Note that it uses the async and await keywords to define asynchronous methods and to call them.

3. In the ready() event method, note that the code begins by getting an Array object that contains all posts. Then, it discards all posts except the first five.

4. In the ready() event method, modify the code so it only gets the posts with ids of 1, 11, 21, 31, and 41. To do that, you can create an array of Promise objects where each promise gets an object by requesting a URL that ends with posts/1, posts/11, posts/21, and so on. Then, you can use the Promise.all() method to execute the asynchronous requests.

5. Run and test the application again to make sure it works as expected. This time, each post should be from a different user.