**Course description:**

This course will give you experience creating HTML interfaces using the JavaScript framework called jQuery. This course is mostly practical based.

The course-taker is required to have JavaScript development skills.

During the course you will be dealing with such tasks as:

* Using jQuery UI widgets to create HTML interfaces
* Using AJAX queries for asynchronous page loading
* Integrating new UI elements into the existing design (jQuery UI theme roller)
* Creating new jQuery plugins
* Templating the client
* [optional] Optimizing the client part of the finished application
* [optional] Developing an interface (requires the argumentation of changes in terms of interface design)

**Course Staff**

****[**training.JavaScript**](mailto:training.JavaScript@dataart.com)

You can always ask questions and receive answers on this mailing list.

**Frequently Asked Questions**

**How long does this course take?**

It takes 80 hours to finish.

# Useful Materials:

1. jQuery:

* Official page with the library manual <http://api.jquery.com/>
* Tips and Tricks <http://stackoverflow.com/questions/182630/jquery-tips-and-tricks>
* Best practices <http://www.smashingmagazine.com/2008/09/16/jquery-examples-and-best-practices/>
* 5 ways to make AJAX calls with jQuery <https://www.edufukunari.com.br/article/5-ways-make-ajax-calls-jquery/>
* Anchor navigation and the Browser’s History

<http://habrahabr.ru/blogs/webdev/31211/>

<http://www.jquerybuzz.com/tag/hashbang+ajax/> ♣

<http://benalman.com/projects/jquery-bbq-plugin/>

* Creating plugins

2. Client side templating: <http://api.jquery.com/category/plugins/templates/> (recommended template)

3. jQuery UI:

* <http://jqueryui.com/demos/> (demos and widget documentations)
* <http://jqueryui.com/themeroller/> (ThemeRoller interface, which helps with the customization of different styles for your library; very useful if you need to quickly integrate widgets into the existing design)

4. JSONP

* <http://www.ibm.com/developerworks/library/wa-aj-jsonp1/>
* <http://api.jquery.com/jQuery.getJSON/>

5. Bower - <http://bower.io/> . To find packages please use <http://bower.io/search/> search packages form.

# TASK

## Project Description:

You will be asked to implement a web interface for an “Internet News” application.

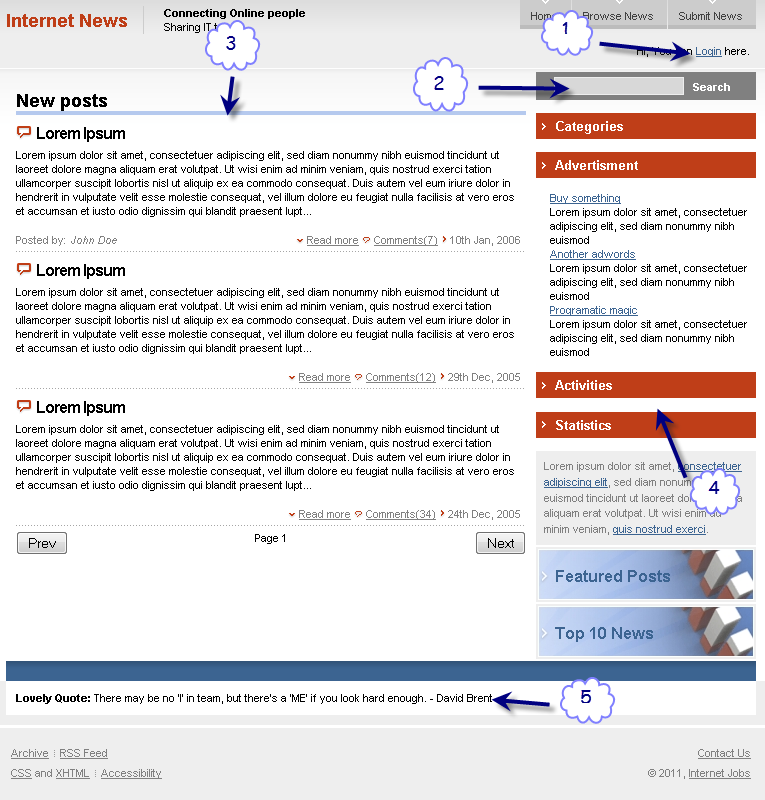
“Internet News” helps users to browse the latest news articles and perform quick searches. This application supports quick registration through AJAX, autocomplete search, AJAX navigation and validation of input data from the user. It all works as a mashup project (provides quotes from a third-party service).

For the design model, a web page design has been developed and an HTML layout is provided (see pic. below)

Interface elements, which you will have to implement, are described in project terms and can be found at the bottom of the picture.

You can find detailed descriptions of every component in the “UI components description” section.

### Mockup



In order to complete this course you will need to implement:

1. **Login dialog box (jQuery Dialog, Validation Plugin)**
2. **Search autocomplete (jQuery Autocomplete)**
3. **Slider-pagination (custom jQuery Plugin Paginator for sliding the news)**
4. **Navigation menu (jQuery Accordion)**
5. **Quote service (JSONP)**
6. **Bower must be used for tracking all plugins/libraries**
7. **Please use Object Oriented Style & Closures as much as you can.**

Write to us at [training.JavaScript](mailto:%20training.JavaScript@dataart.com) when ready or if you have any questions.

• You’re expected to write a client part of a very busy project, so there might be some delays with Ajax data calls. We recommend you to divide the main page load into 3 parts in order to make a better user experience.

1. Static part of page loading (index.html)
2. Initializing the dynamic UI components (Navigation Accordion, Slider-pagination, Search Autocomplete)
3. Data loading from content providers from another domain (Quote Service JSONP)

• Use lazy initialization for every possible component

• You should use existing HTML templates where it’s indicated. You are free to edit the templates due to the different template engine syntax. You also free to use any template engine you want.

• It’s better to use caching everywhere you can. It refers to all AJAX requests. You can use Local Storage to store all the received data.

• Use the loader images for displaying the state of components load (you can use <http://www.ajaxload.info/>)

• Use bower for tracking all the plugins/libraries

• The code should be implemented using the Object Oriented style. JS closures would be an asset.

• The code must be committed to the fabricator’s depository. There’s a Wiki link in the Repository chapter

# Organization of work:

1. Website address is <http://jq-course.local>

2. With this document, you’ll receive the OpenServer pack, configured and operational. Launch the pack and open the website <http://jq-course.local/> to make sure everything’s fine with them. For lauhch you'll have to open OpenServer forlder and run OpenServer x86/x64 depending on your OS. Once you lauch the appropriate .exe file you'll be able to run the server by clicking on tray icon

* + - MySQL was deleted from the environment in order to free up space, so the launch script may display errors. Just ignore them.
    - For launching, stopping or restarting the server you should use the appropriate shortcuts in the root folder.

3. The address of the component’s content distributor called Quote service is http://quote-service.local /api.php

4. After the work is done, you should commit all your changes to Phabricator.

5. We recommend using this template - <http://api.jquery.com/jQuery.template/>

# UI components description:

**• Search autocomplete**: implements the autocomplete function for the search form. In the course of filling the form, the search input must offer variants of words from data source that coincide with the entered characters. Words from data source are to be loaded once during page launch. The form must be submitted automatically with one click on one of the form variants in order to save the user some time. In order to implement that you can use the jQuery UI Autocomplete widget (<http://jqueryui.com/demos/autocomplete/>). Data source : /mocks/search.json.php

**• Navigation menu**: Navigational section menu (on the right side of the screen) must simultaneously display the content of only one section in the form of subsections list. The “Advertisement” section should be opened by default. Block panel’s height must be configurable depending on the filling of the active panel by the content. The actual state of the navigation panel (set before a refresh) must be maintained when you refresh the page(you can use Local Storage or Session Storage to do that). Browser history support: switching the page between the pages using back/forward navigation buttons MUST NOT control the state of the navigation panel. HTML layout is generated dynamically based on the JSON document from the Data source. You can use jQuery UI Accordion (<http://jqueryui.com/demos/accordion/>). Data source: /mocks/navigation.json.php. Template: *templates/nav-cats, templates/nav-subitem.*

**• Login dialog box**: form of login/registration in the overlay (jQuery Dialog). After clicking on the “Login” link in the global navigation, the blocking dialogue box with the login form must appear. After clicking on the “Login” or “Register” buttons, client-side validation must be initiated.

• If the user entered incorrect data (from the validation point of view), the errors messages must be displayed.

If the user is already registered/logged in via Login/Register form, the green greeting text (green text in the place of Login link) should be displayed: “You are logged in as **#email#**”

If the user has just registered – automatically show him “Thanks for the registration. You are logged in as **#email#**” in the informational line of the page (green text in the place of Login link).

Greeting text Template: /templates/login/greet-register-inlay.html.

The user can close the login form at any moment, automatically going back to the page they were viewing.

Login form validation rules:

- Email – valid email

- Password – only letters and numbers.

Registration form validation rules:

- Email – valid email

- Password – only letters and numbers.

- Confirm Password – must match Password field

You can use the jQuery UI Dialog widget (<http://jqueryui.com/demos/dialog/>) in order to create an overlay, and any validation plugin (e.g. <https://jqueryvalidation.org/documentation/> or you can use [Mustache](http://coenraets.org/blog/2011/12/tutorial-html-templates-with-mustache-js/) and [Handlebars](http://handlebarsjs.com/) ).

Template: /templates/login/login-register-overlay.html

**• Slider-pagination**: this component is expected to be developed through one's own individual effort (custom jQuery Plugin). The component for page-to-page navigation through the news list, like on mockup screen (see Mockup section). The page-to-page transition is performed by using the “Next” and “Previous” buttons located at the bottom of the news list. If a corresponding page does not exist, then the button must be disabled. Current page’s number should be displayed between the buttons. The number of news items on the page should be fixed, specified in the config file and equal to 7. Filtering and sorting options are absent. The component’s appearance should be specified in the CSS file. The actual page (the one that the user was browsing prior to refreshing) should be loaded after refreshing the page. Browser history support: there should be an option that allows switching from page to page using the back/forward navigation buttons. History support and remembering the state of a page before a refresh should be implemented through the anchor part of the URL. HTML layout is generated dynamically based on the template and is filled with data from the JSON document in the Data source.

The data to be displayed about each news item:

- Headline

- Body

- Number of comments

- Author Date

(everything else is at the discretion of the course taker)

Data source: /mocks/news-list.json.php

Template: /templates/news-item.html (the template for generating every item in the news list; the wrapper should be created through one's individual effort)

**• Quote service**: while using JSONP you should instance new quotes to the “Lovely Quote” block from the content provider’s page.

NOTE: there may be delays and connection problems.

Data source: <http://quote-service.local/api.php>