Build Engaging Forms Using Core Components and Headless Adaptive Forms on AEM Forms as a Cloud Service {#build-engaging-forms-using-core-components-and-headless}

Lab Overview {#lab-overview}

In this hands-on lab, you learn:

How to use AEM Forms to easily create adaptive forms using the latest core components which are consistent with AEM Sites, enable omnichannel data capture experiences by delivering the adaptive forms as headless forms to web, mobile, and chat. You also learn best practices around styling, customizations, and front-end development.

Key takeaways {#key-takeaways}

- **Business Agility**: As a business user, I can author Form experience for multiple channels easily.
- **Power to frontend developer**: As a frontend developer, I can control the end-user experience using headless forms.
- **Developer Velocity**: As a developer, I can easily and consistently customize Sites and Forms components.

Prerequisites {#prerequisites}

To use this hands on lab:

- Install the latest release of Git. If you are new to Git, see Installing Git.
- Install Node.js 16.13.0 or later. If you are new to Node.js, see How to install Node.js.
- Enable Headless Adaptive Forms for your AEM Forms as a Cloud Service environment.
- Install Microsoft Visual Studio Code or any plain text editor. Examples in document make use of Microsoft Visual Studio Code.

Lesson 1 {#lesson-1}

Objective {#lesson-1-objectives}

Familiarize yourself with AEM Forms as a Cloud Service environment.

Lesson context {#lesson-1-context}

In this lesson, you familiarize yourself with AEM Forms as a Cloud Service environment by navigating the user interface.

Exercise {#lesson-1-excercise}

1. Open your browser and enter the URL of the Cloud Service author environment. For example: https://author-p105303-e986623.adobeaemcloud.com/ui#/aem/aem/start.html
2. Log in to the Cloud Service author environment.
3. To navigate to the AEM Forms UI, click **Forms > Forms & Documents**.

   {width="50%" align="left"}

   {width="50%" align="left"}

   Dismiss any pop-ups related to preferences or information. All the available forms are displayed.

Lesson 2

Objective

Author an adaptive form using the latest core components, configure, and submit the form.

Lesson context

In this lesson, as a business user, you will author an adaptive form for multiple channels like web, mobile, and chat using adaptive forms authoring with standardized OOTB core components for data capture.

Exercise

1. Create a submission endpoint for the form:
   1. Open https://requestbin.com/ in a new browser tab. {width="50%" align="left"}
   2. Click **Create a public bin** and copy the endpoint URL. {width="50%" align="left"}
2. Author an adaptive form using the Wizard interface:
   1. In the browser tab used in Lesson 1, navigate to AEM Forms as Cloud Service web interface and navigate to Forms and Documents.
   2. Click **Create** and select Adaptive Form.
   3. Select the **Blank with Core Components** template from the template selection screen as shown below:
   4. Click the **Style** tab and select the **wknd-theme** theme as shown below:
   5. Click the **Submission** tab and select the **Submit to REST end-point** card and specify the public bin in the **URL for the POST request** field as shown below:
   6. Click **Create**. Specify a name and title to your form. For example, **registration**. Click **Create**.
   7. The adaptive form editor opens. Dismiss any pop-ups or dialogs for preferences or information. Click the components browser on left rail and add the **Header** and **Footer** components respectively to the top and bottom of the blank form.

8. Drag and drop components from the Components browser to create a form, similar to the following:

   {width="50%" align="left"}

3. Add validations to form:

   1. Click the **Phone number** component so that the pop-up menu is displayed. Click the **Wrench icon** in the menu to configure the field.

   2. Open the **validations tab**, mark the field **Required**, and click **Done**. The success message is displayed. {width="50%" align="left"}

      {width="50%" align="left"}

4. Preview and submit the form.

   1. Click **Preview** to preview the form from an end-user perspective.

   2. Fill the form with dummy data.

   3. Submit the form.

   4. In the Request bin tab, check the submitted data.

5. Add interactivity to form with rules:

   1. Click the **Check the box to receive 5% off** component. On the options toolbar, click the Rules icon. The Rule edior option opens.

   2. Create a rule, when the **Check the box to receive 5% off** option is selected, the options for applying credit card is disabled.

6. Publish the form.

   1. Open AEM Forms management interface, for example, `https://author-p105303-e986623.adobeaemcloud.com/ui%23/aem/aem/forms.html/content/dam/formsanddocuments`, and select the form.

   2. Click **Publish**.

      The success message is displayed.

      The publish URL of the form would be similar to `https://publish-p105303-e986623.adobeaemcloud.com/content/forms/af/registration.html`.

   3. To view the published form replace the program ID (pXXXXXX) and environment ID (eXXXXXX) in the above URL with ID's of your environment.

Lesson 3

Objective

Update styles using frontend development best practices.

Lesson context

In this lesson, as a front-end developer, you learn how you can update styling for the previously created adaptive form easily.

Exercise

Set up local repository of the theme:

1. Open the Command Prompt or shell with administrator rights:

   {width="50%" align="left"}

2. On the Command Prompt, use the following command to navigate to **c:\git** folder

   ```
   cd c:\git
   ```

3. Use the following command to clone the theme frontend code:

   ```
   git clone -b WKND https://github.com/adobe/aem-forms-theme-canvas
   ```

4. Use the following command in the listed order to navigate to the **aem-forms-theme-canvas** directory and open Visual Studio Code.

   ```
   cd aem-forms-theme-canvas
   code .
   ```

5. Select **Trust the authors of all files in the parent folder** and click **Yes, I trust the authors**.

   {width="50%" align="left"}

6. To render the form hosted on your cloud service publish environment, rename the `env_template` file. To rename the file, right click the **env_template** file and select the **Rename** option.

   {width="50%" align="left"}

{width="50%" align="left"}

7. Set the following values for the variables in .env file and save the file:

    - **AEM_URL**: Specify your cloud service publish environment. For example, `https://publish-p105303-e986623.adobeaemcloud.com/`
    - **AEM_ADAPTIVE_FORM**: Specify the path of the form. For example, if the form path is `/content/forms/af/registration`, the value of this variable would be `registration`.

      {width="50%" align="left"}

8. In the Command Prompt window, run the following command:

```
npm install
```

> [!NOTE]
>
> - If you get a message asking to update npm via the `npm notice Run npm nstall -g npm@9.6.0`command, ignore the message.
> - Do not run other npm commands unless instructed in the workbook.

9. Now run the following command to preview the form.

```
npm run live
```

Once the above command is executed, wait for the `webpack compiled` message. The form is displayed in a browser tab.

> [!NOTE]
>
> If you experience a blank screen in browser after executing the `npm run live` command for more than 3-4 minutes, change `localhost` in browser URL to 127.0.0.1 and hit **Enter**.

{width="50%" align="left"}

10. In Visual Studio Code, open the `PROJECT\src\site\_variables.scss` file. Notice the `$error` color is a shade of RED.

{width="50%" align="left"}

11. In the browser, submit the form to see the Red color in the **First Name** field.

12. Set the **$error** color to **#5736eb** and save the file.

    {width="50%" align="left"}

13. Refresh the browser and submit the form. Notice error color on the first name field has changed accordingly.

14. In the Command Prompt, press **CTRL+C**, enter **Y**, and press **Enter** key to terminate the npm process. It is important to stop the npm server so it does not conflict with the next set of exercises.

15. Close Visual Studio Code and Command Prompt windows.

Lesson 4

Objective

Render the form to web/mobile and other interfaces as a headless form.

Lesson context

In this lesson, as a frontend developer, you will learn how you can render the adaptive form created previously as a headless form using react spectrum design framework.

Exercise

Setup local repository using react starter project:

1. Open the Command Prompt using administrator rights.

   {width="50%" align="left"}

2. On the Command Prompt, use the following command to navigate to **c:\git** folder

   ```
   cd c:\git
   ```

3. Use the following command to clone the adaptive form react starter project:

   ```
   git clone https://github.com/adobe/react-starter-kit-aem-headless-forms
   ```

4. Use the following commands in the listed order to navigate to the **react-starter-kit-aem-headless-forms** directory and open Visual Studio Code.

```
cd react-starter-kit-aem-headless-forms

code .
```

   The Visual Studio Code window opens.

   {width="50%" align="left"}

To render the form hosted on your cloud service publish environment:

1. Rename the env_template file to .env file. To rename, right-click the **env_template** file and select the **Rename** option.

   {width="50%" align="left"}

2. Set the following values for the variables in the .env file. After updating variables, save the file.

   - **AEM_URL**: Specify the URL of the cloud service publish environment. For example, `https://publish-p105303-e986623.adobeaemcloud.com`
   - **AEM_FORM_PATH**: Specify the path of the adaptive form created in the previous lesson. For example, `/content/forms/af/registration/`

3. Open the command window, ensure you are at the react-starter-kit-aem-headless-forms directory, and run the following command:

```
npm install
```

4. In the Command Prompt window, run the following command:

```
npm start
```

The above command starts a local development server which would render the form definition fetched from AEM in a headless way using the react-spectrum frontend library.

> [!NOTE]
>
> If you experience a blank screen in browser after executing the `npm start` command for more than 3-4 minutes, change `localhost` in browser URL to 127.0.0.1 and hit **Enter**.

Let's check the execution of rules in this headless form:

1. Select the **Check the box to receive 5% off** option. The subsequent option for applying credit card is disabled.

2. Uncheck **Check the box to receive 5% off** to enable the credit card option.

Let's make changes on the form on the server as a business user and view changes reflected in the headless form automatically.

1. Open the AEM Forms management interface in the browser. For example, https://author-p105303-e986623.adobeaemcloud.com/ui#/aem/aem/forms.html/content/dam/formsanddocuments.
2. Select the **contactus** form and click **Edit.** It opens the form in the adaptive forms editor.
3. Select the **Phone number** field and click the **Edit icon (Pencil icon)** in the toolbar. If you are not able to see the pop up toolbar, switch to Edit mode by clicking **Edit** button in top right, left to **Preview** button.

4. Change the label to Mobile Number. Click any empty space in the form and the changes made to the form are saved.

Let's publish the updated form to propagate the changes to publish environment.

1. In the AEM Forms management interface tab, select the registration form, and click **Unpublish**. If you do not see the **Unpublish** button, skip to step 3 to publish the changes directly.
2. Click **Unpublish**. Click **Close** in respective dialog.
3. After the browser refreshes, select the registration form and click **Publish**.
4. Click **Publish**. Click **Close** in the respective dialog.

5. Refresh the browser tab with the headless form displayed. Notice, the Phone number label has changed to Mobile number.

6. Open the Command Prompt window that is used to start the **react-starter-kit-aem-headless-forms** project, press **CTRL+C**, then enter **Y** and press Enter key to terminate the npm process. It is important to stop the npm server so it does not conflict with the next set of exercises.

7. Close Visual Studio Code and Command Prompt windows.

Lesson 5

Objective

Render the form as a headless form using Google Material UI

Lesson context

In this lesson, as a front-end developer, you learn how to render the adaptive form created previously as a headless form using Google Material UI.

Exercise

Setup local repository using Material UI starter project:

1. Open the Command Prompt using administrator rights.

   {width="50%" align="left"}

2. On the Command Prompt, use the following command to navigate to **c:\git** folder:

   ```
   cd c:\git
   ```

3. Run the following commands in the listed order to create a folder named mui and navigate to the mui folder using following commands:

   ```
   mkdir mui

   cd mui
   ```

4. Use the following command to clone the adaptive form react starter project:

```
git clone -b mui-lab https://github.com/adobe/react-starter-kit-aem-
headless-forms
```

5. Use the following command in the listed order to navigate to the **react-starter-kit-aem-headless-forms** folder and open the code in Visual Studio Code:

```
cd react-starter-kit-aem-headless-forms

code .
```

To render the form hosted on your cloud service publish environment:

1. Rename the **env_template** file to **.env** file. To rename, right-click the **env_template** file and select **Rename**.

   {width="50%" align="left"}

2. Set the following values for the variables in the .env file. After updating variables, save the file. Use the **CTRL + S** switch combination to save the file.

   - **AEM_URL**: Specify the URL of the cloud service publish environment. For example, https://publish-p105303-e986623.adobeaemcloud.com
   - **AEM_FORM_PATH**: Specify the path of the adaptive form created in the previous lesson. For example, /content/forms/af/registration/

3. Open the command window, ensure you are at the **react-starter-kit-aem-headless-forms** directory, and run the following command:

```
npm install
```

4. In the Command Prompt window, run the following command:

```
npm start
```

The command starts a local development server and renders the form definition fetched from AEM in a headless way using the Google Material UI frontend library.

> [!NOTE]
>
> If you experience a blank screen in browser after executing the `npm start` command for more than 3-4 minutes, change `localhost` in browser URL to 127.0.0.1 and hit **Enter**.

5. To evaluate the execution of the same business logic in this form rendition:

   Select **Check the box to receive 5% off**. The subsequent option **Would you like to apply for We.Finance Corporate Credit Card Form?** gets disabled.

   {width="50%" align="left"}

Lesson 6

Objective

Create an alternate look and feel of the headless form using Material UI component variations

Lesson context

In this lesson, as a front-end developer, you learn how to create an alternate representation of different components using Material UI for the adaptive form created previously by the business user.

Exercise

Update the variation of components in the headless project. To change the variant of the material UI text input component to `OutlinedInput`:

1. In Visual Code, navigate to the text input component by opening the `index.tsx` file at `src/components/textinput/index.tsx`.
2. Add `//` at the beginning of the code line 103. It converts the line to a comment.

```
//const Cmp = \'outlined\' === appliedCssClassNames ? OutlinedInput: Input;
```

3. Add the following at line 104 to use a different variant of component and save the file. Use the **CTRL + S** switch combination to save the file.

```
const Cmp = OutlinedInput;
```

It is essential to use correct capitalization for 'OutlinedInput' variant else compilation would fail. The local development environment compilation begins automatically in Command Prompt. Wait until you see the following message

```
webpack 5.75.0 compiled with 3 warnings in 6659 ms inside proxy req setting new origin header
```

4. Refresh the browser, if it does not refresh automatically, to see text input component use a different variant.

This change happens for end users without any change to form definition at AEM Forms Server and is specific for the headless channel under consideration. For example, web channel in this lab.

{width="50%" align="left"}

5. Close Visual Studio Code and Command Prompt Windows.

Frequently Asked Questions (FAQs)

+++ Is Adaptive Form wizard available publicly?

Yes, it available with AEM Forms as Cloud Service.

+++

+++ Are core components available publicly?

Yes, Adaptive Forms core components are available with AEM Forms as Cloud Service.

+++

+++ Are Headless forms available publicly?

Yes, Headless forms are available with AEM Forms as Cloud Service.

+++

+++ Do Headless forms require a separate license?

No, Headless forms use the same licensing value metric, number of form submissions.

+++

+++ Are Core components and Headless forms available with AEM 6.5 Forms?

Yes, both adaptive forms core components and headless forms are available with AEM Forms 6.5 Service Pack 16 and onwards.

+++

Next steps

Now that you have learned how to build adaptive forms and deliver them to multiple channels using headless forms, you should try to put your new skills in action. Have fun and go ahead by creating and delivering exceptional data capture experiences to your end users, where they are, at scale!

Resources

- Adaptive Form core components introduction
- Create adaptive form using core components
- Update styling for core component-based AF
- Headless adaptive forms
- Using Headless React starter kit