02/22/2021

Contents

Module 8	Development with Visual Studio Code	3
	Visual Studio Code	4
	VSCode AEM Sync	5
	Exercise 1: Install Visual Studio Code (Local only)	7
	Exercise 2: Install VSCode AEM Sync (Local only)	8
	Exercise 3: Import an AEM project	10
	Exercise 4: Synchronization tools for VSCode	13
	References	17

Module 8

Development with Visual Studio Code

Introduction

Visual Studio Code is a great choice for front-end developers who will primarily be writing CSS/LESS and JavaScript code to create AEM client libraries.

VSCode AEM Sync is an extension for Visual Studio Code to automatically sync changes to files in the Adobe Experience Manager server. It is used for local development.

Objectives

After completing this course, you will be able to:

- Install Visual Studio Code
- Describe VS Integrated Terminal
- Describe VSCode AEM Sync
- Install VSCode AEM Sync
- Test VSCode AEM Sync Extension

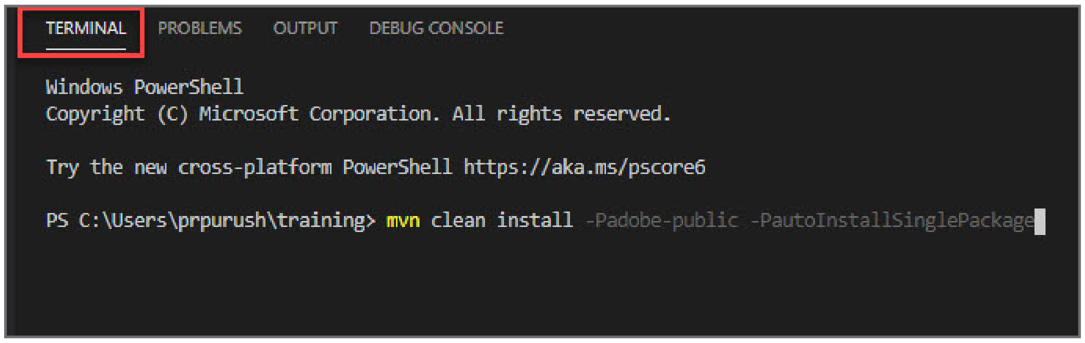
Visual Studio Code

Visual Studio Code is a free source-code editor developed by Microsoft for Windows, Linux and macOS. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages (such as C++, C#, Java, Python, PHP, and Go) and runtimes (such as .NET and Unity). Visual Studio Code also includes support for debugging, embedded Git control and GitHub, syntax highlighting, intelligent code completion, snippets, and code refactoring.

VS Integrated Terminal:
In Visual Studio Code, you can open an integrated terminal, initially starting at the root of your workspace. This can be convenient as you do not have to switch windows or alter the state of an existing terminal to perform a quick command-line task.

To open the terminal:

- Use the **Ctrl+`** keyboard shortcut with the backtick character.
- Use the **View** > **Terminal** menu command.
- From the Command Palette (Ctrl+Shift+P), use the View: Toggle Integrated Terminal command.



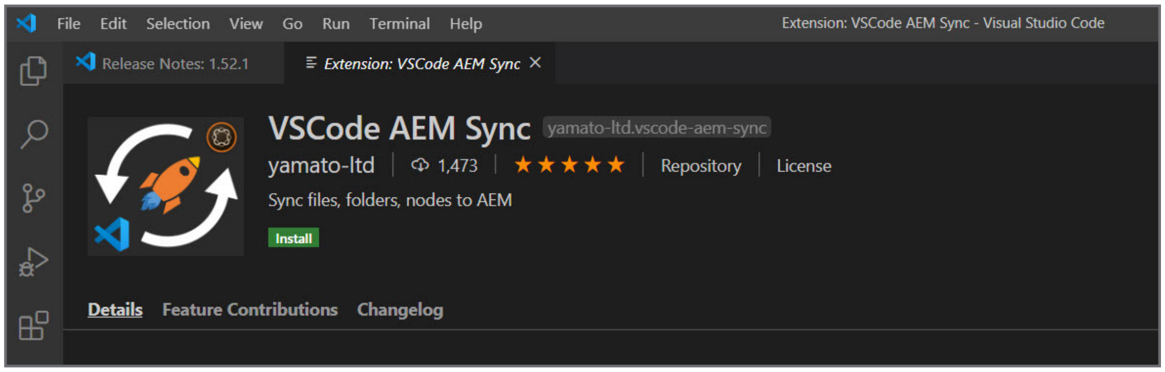
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\prpurush\training> mvn clean install -Padobe-public -PautoInstallSinglePackage
```

VSCode AEM Sync

An extension for Visual Studio Code that allows AEM Developer to sync their code updates smoothly. You can sync not only files also .content.xml, dialog.xml, etc., can be synced by just one click. This sync feature is ported from AEM Brackets Extension.


File Edit Selection View Go Run Terminal Help
Extension: VSCode AEM Sync - Visual Studio Code
Release Notes: 1.52.1
Extension: VSCode AEM Sync
VSCode AEM Sync
yamato-ltd.vscode-aem-sync
yamato-ltd
1,473
Repository
License
Sync files, folders, nodes to AEM
Install
Details
Feature Contributions
Changelog

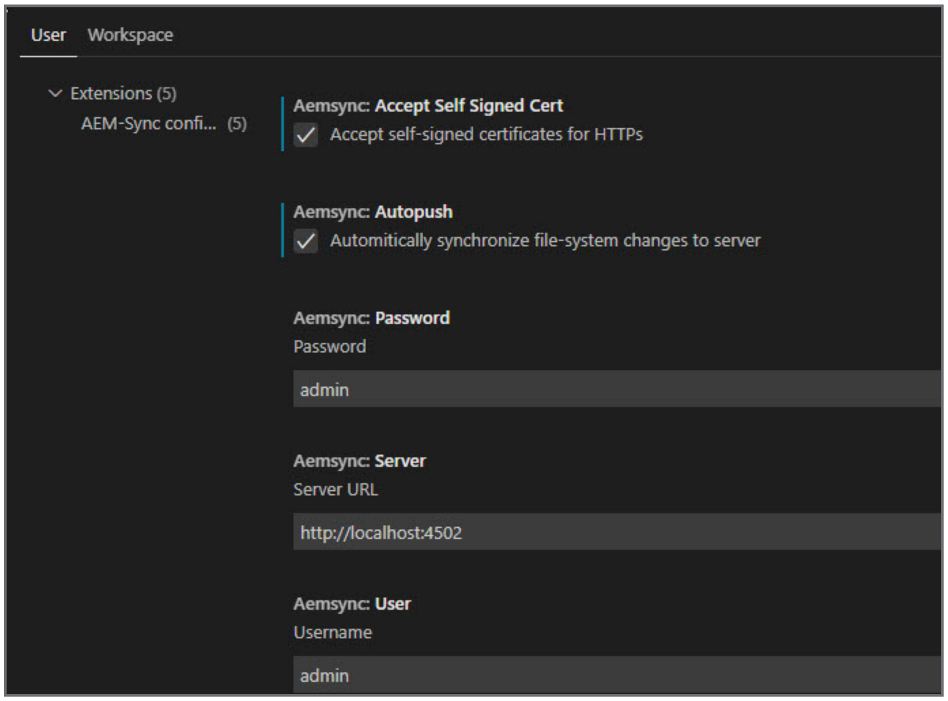
Settings

The menu command File > Preferences > Settings (Code > Preferences > Settings on Mac) provides entry to configure user and workspace settings.

The following settings can be configured:

- aemsync.server - Server URL
- aemsync.user - Username
- aemsync.password - Password
- aemsync.autopush - Automatically synchronize file-system changes to server
- aemsync.acceptSelfSignedCert - Accept self-signed certificates for HTTPs

If auto-push feature is enabled, Export to AEM Server is executed automatically, when you save.



User Workspace

Extensions (5)
AEM-Sync confi... (5)

Aemsync: Accept Self Signed Cert
✓ Accept self-signed certificates for HTTPs

Aemsync: Autopush
✓ Automitically synchronize file-system changes to server

Aemsync: Password
Password
admin

Aemsync: Server
Server URL
http://localhost:4502

Aemsync: User
Username
admin

Exercise 1: Install Visual Studio Code (Local only)

In this exercise, you will install Visual Studio Code.

Note: You can skip this exercise if you use a ReadyTech environment because AEM is already installed as part of the image.

1. Download the latest VS Code installer, for your operating system using the link below:
   https://code.visualstudio.com/download/

Note: If you do not have the ability to download Visual Studio from the Internet, you can use the installer in the distribution-files folder provided to you by your instructor.

2. Install Visual Studio Code.

- **Windows**
  - › Double-click on the .exe file, for example VSCodeUserSetup-ia32-1.38.1.exe, to start the installation and follow the instructions in the installer.
- **Mac OSX**
  - › Double-click on the .zip file, for example VSCode-darwin-stable.zip, to start the installation and follow the instructions in the installer.

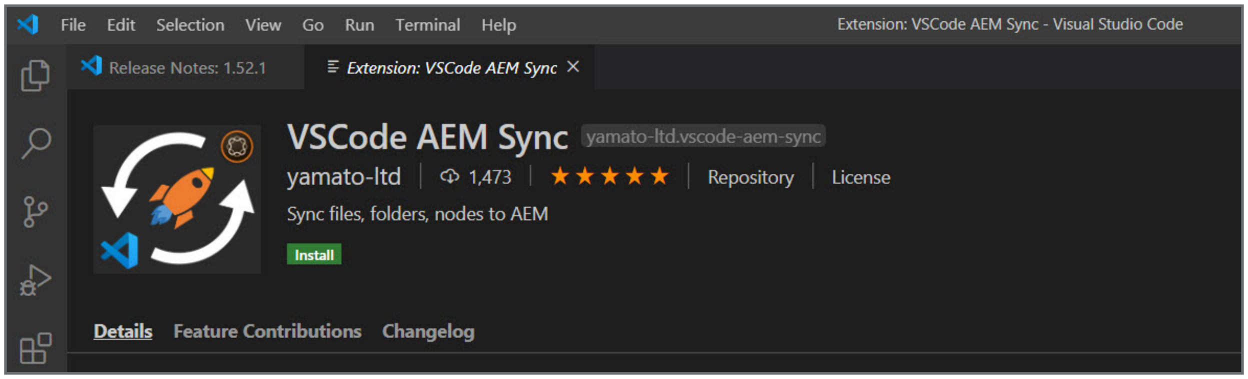
Congratulations! You have installed Visual Studio Code and are now ready to install the VSCode AEM Sync extension.

Exercise 2: Install VSCode AEM Sync (Local only)

In this exercise, you will install VSCode AEM Sync.

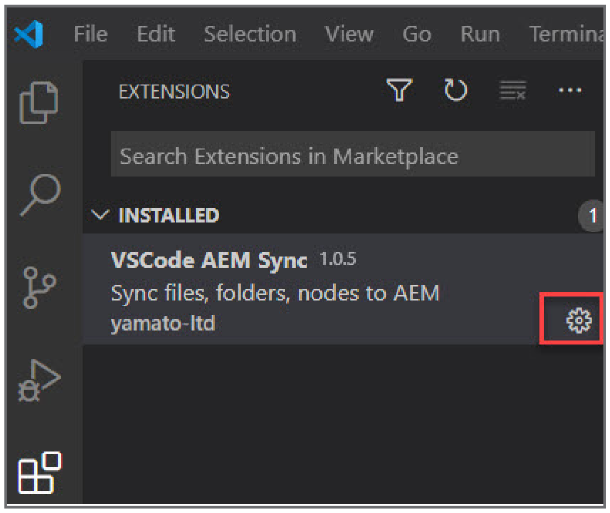
Note: You can skip this exercise if you use a ReadyTech environment because AEM is already installed as part of the image.

1. Open Visual Studio Code.
2. In a browser, go to https://marketplace.visualstudio.com/items?itemName=yamato-ltd.vscode-aem-sync
3. Click the **Install** button. **VSCode AEM Sync** Readme will open in the Visual Studio Code window, as shown:



4. Click **Install** to install the extension.
5. In **Visual Studio Code**, select **View** > **Extensions** from the menu bar. The **VSCode AEM Sync Extension** is displayed and enabled in the **EXTENSIONS** area.

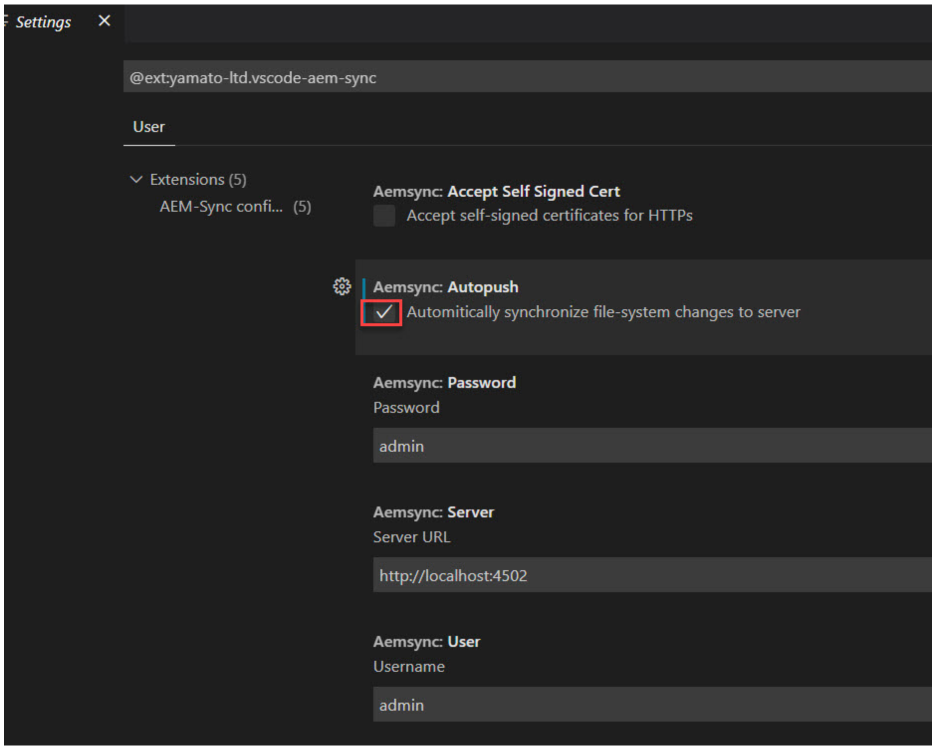
6. Click **VSCode AEM Sync** and click the Manage icon (gear icon), as shown:



VSCode AEM Sync 1.0.5
Sync files, folders, nodes to AEM
yamato-ltd

7. In the Menu, click **Extension Settings.** The **Settings** tab opens.

8. Verify that the extension is properly configured, as shown:
   › Aemsync: Autopush - Enabled
   › Password
   › Server
   › User



@ext:yamato-ltd.vscode-aem-sync

Aemsync: Accept Self Signed Cert
Accept self-signed certificates for HTTPs

Aemsync: Autopush
✓ Autimotically synchronize file-system changes to server

Aemsync: Password
Password
admin

Aemsync: Server
Server URL
http://localhost:4502

Aemsync: User
Username
admin

9. Close the **Settings** tab by clicking **X**.

You have installed the VSCode AEM Sync Extension for Visual Studio Code.

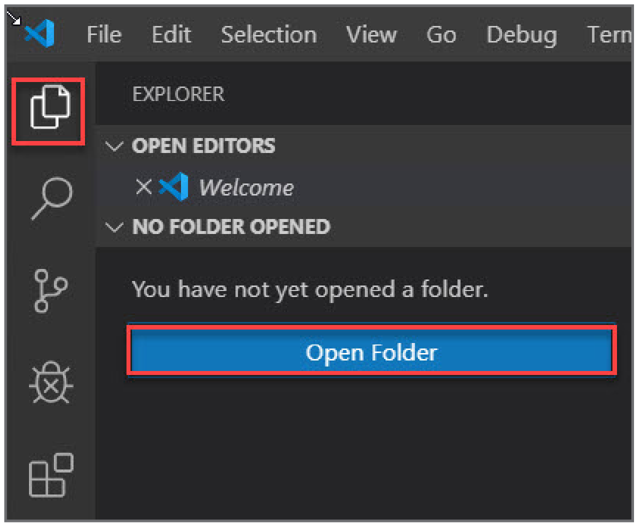
Exercise 3: Import an AEM project

In this exercise, you will import an AEM project into Visual Studio Code.

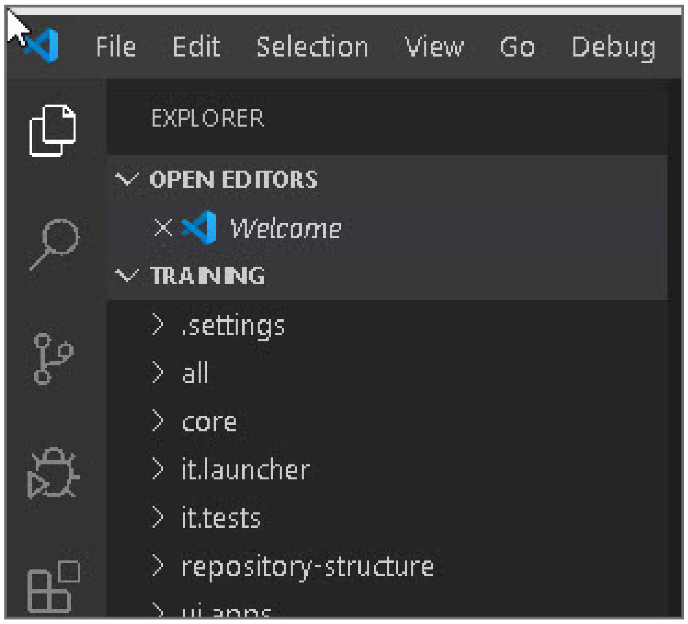
Note: This exercise assumes you have a local Maven project. <AEM Project Directory> represents the location of this local Maven project.

Note: You can skip this task if you have already installed your AEM project.

1. Confirm that your AEM service is running.
2. In **Visual Studio Code**, click the Explorer icon on the left (OR you can press Ctrl+Shift+E) and click **Open Folder**, as shown:

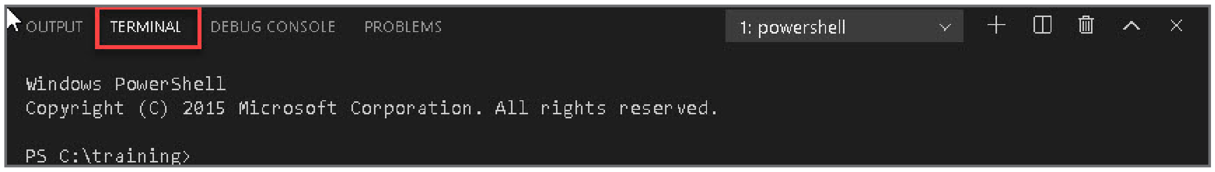


3. Navigate to **<AEM Project Directory>** and click **Select Folder** to open. The folder opens in **Visual Studio Code**, as shown:



To use the Integrated Terminal:

4. Select the **Terminal** tab in the window at the bottom, as shown.

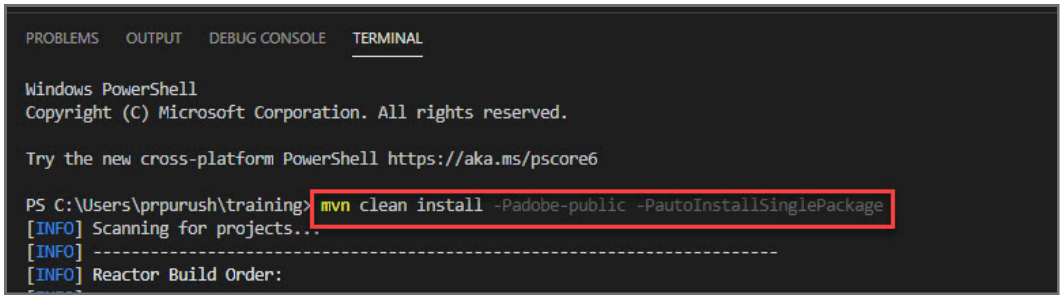


```
Windows PowerShell
Copyright (C) 2015 Microsoft Corporation. All rights reserved.

PS C:\training>
```

5. You can deploy your AEM project from the Integrated Terminal, as shown:

```
$mvn clean install -Padobe-public -PautoInstallSinglePackage
```

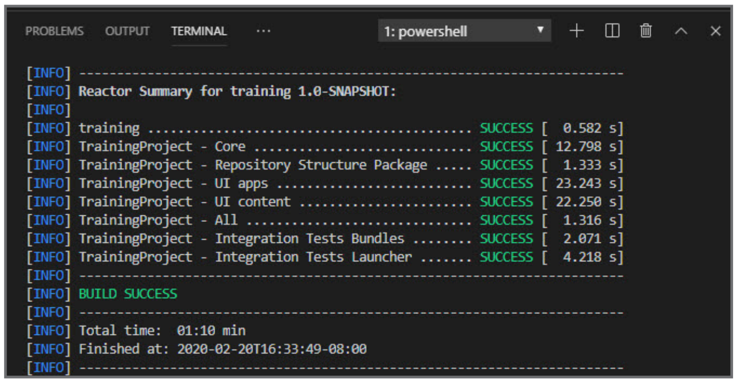


```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\prpurush\training> mvn clean install -Padobe-public -PautoInstallSinglePackage
[INFO] Scanning for projects...
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Build Order:
```

6. Verify the install is successful, as shown:



```
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary for training 1.0-SNAPSHOT:
[INFO]
[INFO] training ........................................... SUCCESS [  0.582 s]
[INFO] TrainingProject - Core ............................. SUCCESS [ 12.798 s]
[INFO] TrainingProject - Repository Structure Package ..... SUCCESS [  1.333 s]
[INFO] TrainingProject - UI apps .......................... SUCCESS [ 23.243 s]
[INFO] TrainingProject - UI content ....................... SUCCESS [ 22.250 s]
[INFO] TrainingProject - All .............................. SUCCESS [  1.316 s]
[INFO] TrainingProject - Integration Tests Bundles ........ SUCCESS [  2.071 s]
[INFO] TrainingProject - Integration Tests Launcher ....... SUCCESS [  4.218 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  01:10 min
[INFO] Finished at: 2020-02-20T16:33:49-08:00
[INFO] ------------------------------------------------------------------------
```

Congratulations! You have successfully imported an AEM project.

Exercise 4: Synchronization tools for VSCode

In this exercise, you will verify auto push is enabled and sync the AEM content.

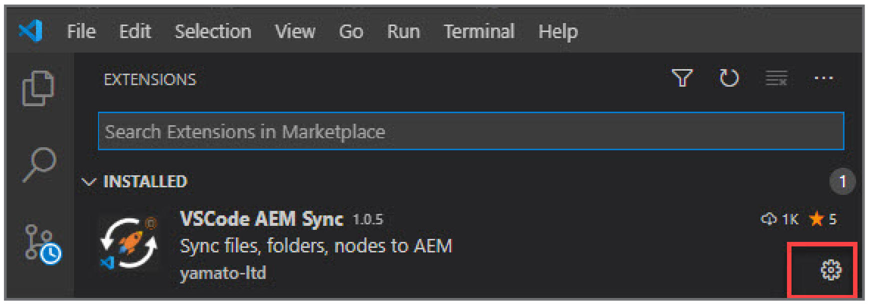
Note: Make sure you have Visual Studio already installed and an AEM author service running locally on port 4502. This exercise already assumes you have a Maven project created from the latest archetype. If you do not have any of these things, refer to earlier sections of this guide.

This exercise consists of the following tasks:

1. Verify auto push is enabled
2. Sync AEM Content

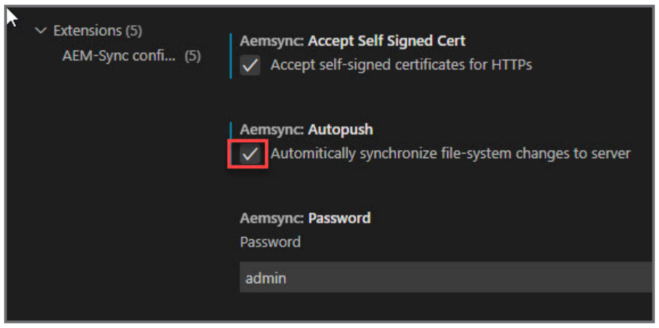
Task 1: Verify auto push is enabled

1. In **Visual Studio Code**, select **View** > **Extensions** from the menu bar. The **VSCode AEM Sync Extension** is displayed and enabled in the **EXTENSIONS** area.
2. Click **VSCode AEM Sync** and click the Manage icon (gear icon), as shown:



VSCode AEM Sync 1.0.5
Sync files, folders, nodes to AEM
yamato-ltd

3. In the Menu, click **Extension Settings.** The **Settings** tab opens.
4. Verify **Autopush** is enabled, as shown:



Aemsync: **Accept Self Signed Cert**
✓ Accept self-signed certificates for HTTPs

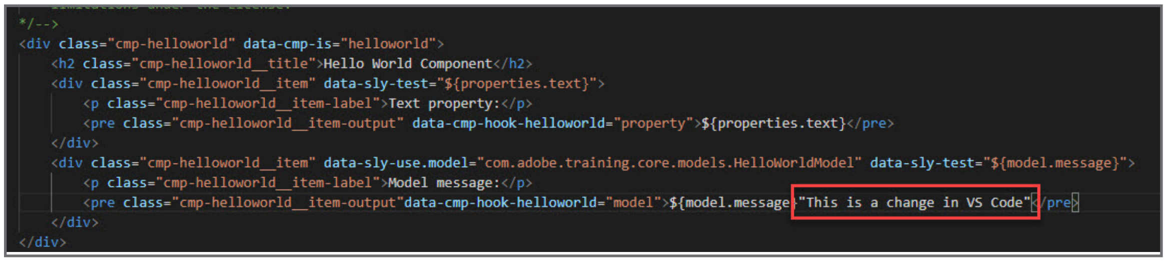
Aemsync: **Autopush**
✓ Automitically synchronize file-system changes to server

Aemsync: **Password**
Password
admin

Task 2: Sync AEM Content

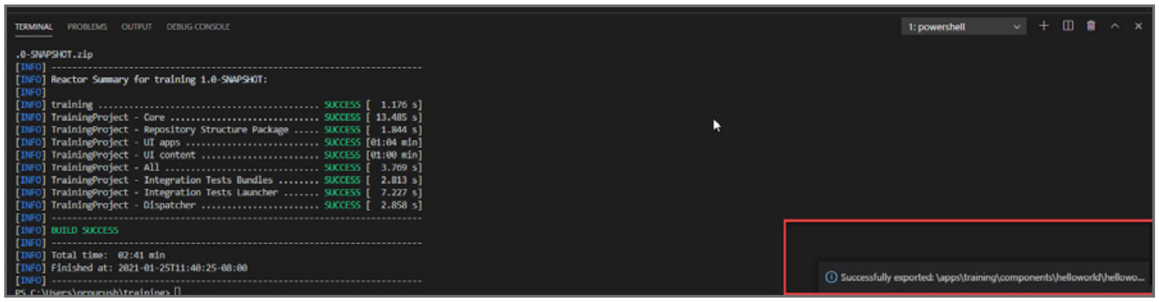
If your project doesn't have the Helloworld component, select a component of your choice to complete this exercise.

1. In VS Code Explorer, navigate to: **<AEM project>.ui.apps** > **src / main /content/jcr_root > apps> training > components > helloworld**.
2. Double-click **helloworld.html**. The HTML page opens.
3. Add the following line at the end of the **<pre>** tag in the code: "**This is a change in VS Code**" .



```
*/-->
<div class="cmp-helloworld" data-cmp-is="helloworld">
    <h2 class="cmp-helloworld__title">Hello World Component</h2>
    <div class="cmp-helloworld__item" data-sly-test="${properties.text}">
        <p class="cmp-helloworld__item-label">Text property:</p>
        <pre class="cmp-helloworld__item-output" data-cmp-hook-helloworld="property">${properties.text}</pre>
    </div>
    <div class="cmp-helloworld__item" data-sly-use.model="com.adobe.training.core.models.HelloWorldModel" data-sly-test="${model.message}">
        <p class="cmp-helloworld__item-label">Model message:</p>
        <pre class="cmp-helloworld__item-output"data-cmp-hook-helloworld="model">${model.message}"This is a change in VS Code"</pre>
    </div>
</div>
```

4. Save the changes.
5. Verify the success message "Successfully exported: \apps\<AEM project>\components\.." at the bottom of the screen, as shown:

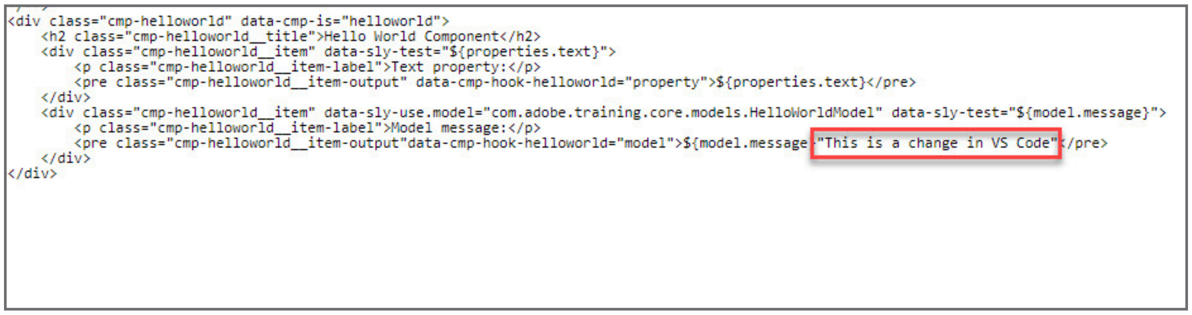


```
.0-SNAPSHOT.zip
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary for training 1.0-SNAPSHOT:
[INFO]
[INFO] training ........................................... SUCCESS [  1.176 s]
[INFO] TrainingProject - Core ............................. SUCCESS [ 13.485 s]
[INFO] TrainingProject - Repository Structure Package ..... SUCCESS [  1.844 s]
[INFO] TrainingProject - UI apps .......................... SUCCESS [01:04 min]
[INFO] TrainingProject - UI content ....................... SUCCESS [01:00 min]
[INFO] TrainingProject - All .............................. SUCCESS [  3.769 s]
[INFO] TrainingProject - Integration Tests Bundles ........ SUCCESS [  2.813 s]
[INFO] TrainingProject - Integration Tests Launcher ....... SUCCESS [  7.227 s]
[INFO] TrainingProject - Dispatcher ....................... SUCCESS [  2.858 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  02:41 min
[INFO] Finished at: 2021-01-25T11:40:25-08:00
[INFO] ------------------------------------------------------------------------
PS C:\Users\prgurush\training>
```

Successfully exported: \apps\training\components\helloworld\hellowo...

Note: When you save helloworld.html, the AEM Server connection in Visual Studio Code automatically exports helloworld.html to the JCR.

6. In **CRXDE Lite**, browse to **apps** > **training** > **components** > **content** > **helloworld** > **helloworld.html** and double-click **helloworld.html.** The HTML page opens.
7. Verify the change in CRXDE Lite:



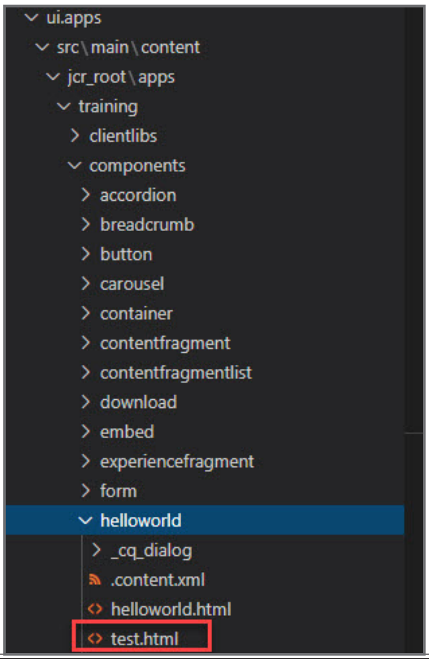
```
<div class="cmp-helloworld" data-cmp-is="helloworld">
    <h2 class="cmp-helloworld__title">Hello World Component</h2>
    <div class="cmp-helloworld__item" data-sly-test="${properties.text}">
        <p class="cmp-helloworld__item-label">Text property:</p>
        <pre class="cmp-helloworld__item-output" data-cmp-hook-helloworld="property">${properties.text}</pre>
    </div>
    <div class="cmp-helloworld__item" data-sly-use.model="com.adobe.training.core.models.HelloWorldModel" data-sly-test="${model.message}">
        <p class="cmp-helloworld__item-label">Model message:</p>
        <pre class="cmp-helloworld__item-output"data-cmp-hook-helloworld="model">${model.message "This is a change in VS Code"</pre>
    </div>
</div>
```

8. Next, we will make a change in CRXDE Lite and import it back into our Visual Studio Code Project. In **CRXDE Lite**, navigate to **apps** > **training** > **components** > **helloworld**.
9. Right-click the **helloworld** component node, and select **Create > Create File**.
10. Enter the name as **test.html.**
11. Click **OK**. The file is created. Click **Save All** in the upper left to save the changes.

Note: Ensure to click Save All after every change in CRXDE lite.

12. In **Visual Studio Code**, under **<AEM project>.ui.apps**, select **/src/main/content/jcr_root/apps/training/components/helloworld**, right-click and select **Import from AEM Server.**
13. Verify the success message "Successfully imported: \apps\<AEM project>\components\..". The selected node and its children are imported from the server.

14. Under **<AEM project>.ui.apps**, navigate to **/apps/training/components/helloworld** and verify **test.html** appears in your project, as shown:



```
ui.apps
  src\main\content
    jcr_root\apps
      training
        clientlibs
        components
          accordion
          breadcrumb
          button
          carousel
          container
          contentfragment
          contentfragmentlist
          download
          embed
          experiencefragment
          form
          helloworld
            _cq_dialog
            .content.xml
            helloworld.html
            test.html
```

Tip: If you create something in CRXDE Lite that you want to keep, you must sync it back to **Visual Studio Code** using the process above.

References

You can use the following links for more information on:

- https://code.visualstudio.com/download
- https://code.visualstudio.com/docs/?dv=win32user
- https://marketplace.visualstudio.com/items?itemName=yamato-ltd.vscode-aem-sync