**Practical Exercise 24 - Setting Up Our JavaScript Custom Action**

**Exercise Description**

**In this practical exercise, our goal is to explore how to setup a JavaScript custom action in GitHub Actions.**

The goal of our JavaScript custom action is to check for updates in the npm dependencies of a project, and to abstract that behind an easy-to-use reusable action. Here are the instructions for the exercise:

1. Create the folder .github/actions/js-dependency-update. This folder is where we will host all the files for our JavaScript custom action.
2. Open a terminal and change into this directory.
3. Initialize an npm project by running the command npm init -y.
4. Install the necessary dependencies by running npm install @actions/core@1.10.1 @actions/exec@1.1.1 @actions/github@6.0.0 --save-exact.
5. Create a file named action.yaml under the folder .github/actions/js-dependency-update.
6. In the action.yaml file, add the following properties:
   1. A name of Update NPM Dependencies;
   2. A description of "Checks if there are updates to NPM packages, and creates a PR with the updated package\*.json files";
   3. Add a top-level runs key. This is the core of defining our JavaScript custom action. For a JavaScript custom action, the runs key has the following shape:
      * 1. runs:
        2. using: node20
        3. main: index.js

where:

* + 1. using: <Node version> defines with which version of NodeJS the action will be run.
    2. main: <JavaScript file> defines which file will be executed as the entrypoint of our JavaScript custom action.

1. Create an index.js file under the folder .github/actions/js-dependency-update and add the following code to the file:
   1. const core = require('@actions/core');
   3. async function run() {
   4. core.info('I am a custom JS action');
   5. }
   7. run();

The code above is leveraging the @actions/core package to write a line to the output of our custom action. In the next exercise we will continue the development of the JavaScript code.

1. Create a file named 17-2-custom-actions-js.yaml under the .github/workflows folder at the root of your repository.
2. Name the workflow 17 - 2 - Custom Actions - JS.
3. Add the following triggers with event filters and activity types to your workflow:
   1. workflow\_dispatch
4. Set the run-name option of the workflow to 17 - 2 - Custom Actions - JS.
5. Add a single job named dependency-update to the workflow.
   1. It should run on ubuntu-latest.
   2. It should contain two steps:
      1. The first step should checkout the code.
      2. The second step, named Check for dependency updates, should use the recently created JS custom action. To reference a custom action created in the same repository as the workflow, you can simply provide the path of the directory where the action.yaml file is located. In this case, this would be ./.github/actions/js-dependency-update.
6. Modify the .gitignore file at the root of the repository so that the node\_modules folder under the js-dependency-update folder is **not** ignored. It should be committed if we want our action to work correctly. One way of doing that is to add the following like to the file: !.github/actions/\*\*/node\_modules. This will make sure that all node\_modules folders under any subdirectory of the .github/actions folder are committed, while still ignoring the node\_modules directories from other folders.
7. Commit the changes and push the code. Trigger the workflow from the UI and take a few moments to inspect the output of the workflow run.