**Practical Exercise 02 - Using Different Events to Trigger Workflows**

**Exercise Description**

**In this practical exercise, our goal is to explore the different ways we can trigger workflows in GitHub Actions.**

Here are the instructions for the exercise:

1. Create a file named 02-workflow-events.yaml under the .github/workflows folder in the root of your repository.
2. Name the workflow 02 - Workflow Events.
3. Add the following triggers to your workflow:
   1. push
4. Add a single job to the workflow:
   1. The job, named echo, should run on ubuntu-latest and contain a single step, named Show the trigger, which prints the type of the name of the event that triggered the workflow.
5. Commit the changes and push the code. Take some time to inspect the output of the workflow run.
6. Now add more triggers to the workflow:
   1. pull\_request
   2. schedule (cron expression)
   3. workflow\_dispatch
7. Commit the changes and push the code. Take some time to inspect the different ways the workflow is triggered.
   1. You can create a pull request on GitHub to see how this changes the output of the workflow run.
   2. Also give it a try to trigger it from the UI. To do so:
      1. Click under the "Actions" tab in the home page of the repository.
      2. Select the workflow named 02 - Workflow Events on the left of the screen.
      3. Click on the "Run workflow" button on the right side of the screen, next to the message "This workflow has a workflow\_dispatch event trigger."
8. After exploring the different ways to trigger a workflow, reduce the list of triggers to leave only workflow\_dispatch to prevent this workflow from running with every push and pollute the list of workflow runs.

**Tips**

**Using a valid cron syntax**

At the time of this recording, GitHub Actions does not support cron job definitions containing six elements (for example, '0 0 \* \* \* \*'), only definitions containing five elements. Check the resources section of this lecture for a cron generator that uses the valid syntax.

To define a trigger using cron, you should use the following syntax:

1. on:
2. schedule:
3. - cron: '<cron expression>'

**Accessing the name of the event that triggered the workflow**

To access the name of the event triggering the workflow, you can use the following special syntax: ${{ github.event\_name }}. For example:

1. steps:
2. - name: Event name
3. run: |
4. echo "Event name: ${{ github.event\_name }}"

We explore the github context in **Section 7: Using Contexts**, and the ${{ }} syntax in **Section 08: Expressions**.

Resources for this lecture: <https://crontab.cronhub.io/>