Setting up Azure container registry task to automatically pull and build upstream image

Azure is providing this ACR Task feature(Still in preview) that can hook up with registry to achieve some great functionalities. Its not feature rich within GUI at this moment of this writing so we have to use azure-cli to configure it.

With DockerHub begins to limit anonymous pull, we start to observe pipeline build failure due to dependency of external images. ACR Task can help here. With the help of ACR Task, we can get some very smart solution where any update to the dependent upstream image will get its downstream image build to keep it in sync. For example, if you have a Dockerfile written like this:

hello-world.dockerfile

```
FROM rwang/hello-world:latest
```

With an ACR Task config like below

hello-world-task.yaml

```
version: v1.1.0
steps:
  - build: -t $Registry/hello-world:$ID -t $Registry/hello-world:latest -f hello-world.dockerfile .
    pull: true
  - push:
    - $Registry/hello-world:$ID
    - $Registry/hello-world:latest
```

Azure will automatically monitor any changes in the upstream **rwang/hello-world:latest** image. Once it changes, an automatic build (By default, Task run fires hourly) will kick off and push the built image into your ACR registry. If you have multiple upstream image specified in dockerfile(multiple **FROM** keywords, such as in a multi stage scenario), only the last one will be honored.

How to setup Task config

You need to have a repo setup which contains the Dockerfile and ACR Task config file as shown above. If you are using a private git repo, an additional access token is needed in order to let ACR Task be able to read the files in it. Here I use my personal github repo which is publicly accessible, so no access token is needed here.

This will provision a task named **hello-world** within the ACR called **pharmacy**, which monitors the dockerfile specified in **hello-world-task.yaml** from the personal public github repo, and won't be triggered on either commit or pull request creation. Which means only the upstream image change will trigger the task.

```
az acr task create -n hello-world -r pharmacy --context https://github.com/deanmax/acr-task-examples.git --
commit-trigger-enabled false --pull-request-trigger-enabled false -f hello-world-task.yaml
```

Note that you'll need to manually trigger the run for the first time to enable the automatic monitoring. To do that

```
az acr task run hello-world
```

Now you have your first ACR Task setup to automatically sync up your customized image with the upstream. Update your CI pipeline and start using this customized image to replace the upstream one!

Task run logs can be found on the ACR GUI or **az acr task** CLI interface.

References

- ACR Task overview - https://docs.microsoft.com/en-us/azure/container-registry/container-registry-tasks-overview
- ACR task configuration file scheme - https://docs.microsoft.com/en-us/azure/container-registry/container-registry-tasks-reference-yaml
- Samples of supported ACR Tasks - https://docs.microsoft.com/en-us/azure/container-registry/container-registry-tasks-samples